DESIGN TOOL FOR SIMULATION OF NANOCOMPOSITE MATERIAL PROPERTIES

A Thesis
Presented to
The Academic Faculty

by

Johnny Lee Worthy III

In Partial Fulfillment
of the Requirements for the
Research Option in the
School of Aerospace Engineering

Georgia Institute of Technology
February 2013

COPYRIGHT 2013 BY JOHNNY LEE WROTHY III

DESIGN TOOL FOR SIMULATION OF NANOCOMPOSITE

MATERIAL PROPERTIES

Approved by:

Dr. Raghu Pucha, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Lakshmi Sankar
School of Aerospace Engineering
Georgia Institute of Technology

Date Approved: 22 April 2013

Table of Contents

List of Figures

List of Tables

ACKNOWLEDGEMENTS

I would like to thank Dr. Raghu Pucha for advising and guiding my research over the years. I would also like to thank Dr. Kyriaki Kalaitzidou and Dr. Satish Kumar for their useful technical discussions. The contributions of the members of Dr. Kalaitzidou's lab in the Mechanical Engineering department are greatly appreciated. The work of the undergraduate members, from Georgia Tech and also from visiting universities, of Dr. Pucha's research group over the past years is acknowledged and appreciated. Finally, I would like to thank Summer Undergraduate Research Experience (SURE), President's Undergraduate Research Award (PURA), and the Peach State Louis Stokes Alliance for Minority Participation (PS-LSAMP) for providing financial support for my research.

SUMMARY

Manufacturing of nanofiller composites is one of the rapidly evolving areas of composite materials research due to the unique properties exhibited by nanoscale reinforcements. Carbon nanotubes (CNTs), for example, have superior mechanical properties with a Young's Modulus on the order of 1 TPa. However, there exists a considerable gap between the individual properties of CNTs and their stress transfer efficiency in composites. Computational modeling techniques for the determination of mechanical properties of nanocomposites have proven to be very effective through parametric studies to facilitate the design and development of nanocomposites. Integrated design and analysis tools for composite materials with nanofiller are presented in this thesis. These tools combine statistical image analysis with 3D modeling to automate the creation of a Representative Volume Element (RVE) models capable of simulating the mechanical and electrical properties of nanocomposites with various filler morphology and process parameters.

CHAPTER I

1.1 Introduction

Carbon nanotubes have been the focus of much of the research in the nanotechnology area since their discovery. With favorable mechanical properties, such as a tensile strength of over 100 times that of steel and extremely high electrical and thermal conductivity, the use of these carbon nanotubes in polymer composites has been an area of interest. In certain applications in which structural materials need to be electrically conductive or need to dissipate static, the use of metallic materials may be unfeasible due to the increased weight. An electrically conductive polymer can provide the same electrical properties as a metallic material while being lightweight, making them favorable for applications such as aerospace where it is necessary to dissipate static buildup. If the use of metallic materials for static dissipation can be replaced with conducting polymers, the equivalent performance coupled with a lighter material overall would have a profound effect on the aircraft industry. With this in mind, it is important to study the properties of these electrically conductive polymers. Experimental work has been done to determine the percolation threshold of carbon nanotubes inside a polymer block concluding that the conductivity of a polymer increases substantially until this percolation threshold has been reached. Ounaies determined the percolation threshold of SWNTs of length 3000nm with uniform dispersion to be at .05% volume fraction [1]. Mdhari determined the percolation of MWNTs with diameters between 30-50nm and lengths of 5000nm to be at approximately 1.2% volume fraction [2]. Given these values for percolation were achieved experimentally, following work in the field of percolation went to analytically and computationally solving for the percolation of a system. The

Newman and Ziff algorithm for percolation is one that is highly adopted and used in this field as it is efficient, utilizing a union/find algorithm to determine if a path across the polymer block has been formed by the network of carbon nanotubes [3]. From a computational modeling approach, there has been research in which methods are developed where a network of carbon nanotubes is generated considering some of the parameters such as waviness, agglomeration, dispersion, etc. In this work Representative Volume Element (RVE) based three-dimensional (3D) models with various nanofiller geometries and process parameters are presented for design and analysis of composite materials. Analytical, CAD and CAE tools are integrated to develop user interface tools with automated 3D models for mechanical and electrical analysis. Various process parameters in manufacturing nanocomposites are quantified using image analysis techniques. A filler-to-filler distance algorithm is incorporated in developing 3D network of fillers within matrix RVE to account for filler-filler interactions and compatibility. Stress-strain behavior of metal matrix nanocomposites, effective modulus and electrical conductivity of polymer nanocomposite fibers are presented as case studies to demonstrate the capabilities of developed RVE-based design and analysis tools. The unique, automated design and analysis framework presented integrates various software tools, quantifies the effect of process parameters of experimental composites with nanofillers, and provides quick what-if analysis for manufacturing application specific composites.

CHAPTER II

2.1. Parameterization of Filler Morphologies

Filler materials for nanocomposites typically assume one of three morphologies: spherical, disk, or fiber shaped. In order to create a comprehensive tool capable of modeling all three morphologies, the parameters which control the shapes of the three different morphologies had to be considered within the scope of manufacturing. It was determined that the code needed to be able to replicate the various states that a filler may have in an experimental composite, thus parameters were defined. The modeling methodology focuses on parameterization because with the proper parameter set defined for an RVE, the better correlation between the simulated geometry and the experimental geometry on which it is based.

2.1.1. Dispersion

The dispersion of the filler within the RVE is one of the most basic parameters controlled by the algorithm. The dispersion is defined as the location of the center of mass of the filler particle. The ability to control this allows for the phenomenon of agglomeration to be simulated. Agglomeration is the clustering of particles within the RVE due to poor mixing or to van der Waals interactions at the molecular level during manufacturing.

2.1.2. Orientation

It has been found experimentally that the orientation of the particles has a major impact on properties of the composite [8-10]. Fibers aligned in the loading direction offer much improved tensile properties over fibers randomly oriented in an RVE. Thus, orientation was considered to be an important parameter for the algorithm. The orientation is defined with two components, the in-plane orientation, θ, and the out-of-plane orientation, φ. The

RVE models were constructed such that the x axis is the longitudinal axis of the model. Thus for θ = φ = 0, the longitudinal axes of the fibers are aligned with the x axis. For disk shaped fillers, θ = φ = 0 corresponds to the disk surface faces being normal to the x axis. For spherical shaped fillers, this parameter is ignored.

2.1.3. Filler Geometry

Spheres and Disks

Each of the filler shapes has different parameters for their geometry definition. Spherical fillers are only defined geometrically by their radius, *r*. Disk shaped particles are defined by a radius, *r*, and a thickness, *t*. For disk shaped particles, these two parameters can be combined to define the aspect ratio, *ar*, of the disk where $\mathbf{ar} = \frac{\mathbf{2r}}{\mathbf{t}}$. To define a disk geometry, only 2 of the three parameters are needed. Disk shaped particles, such as exfoliated graphite nanoplatelets (xGNPs), are often not flat disks, but warped disks. However, due to the complexity of defining this geometry and the very large value of *ar*, all disk shaped fillers are modeled as flat disks.

Fibers

The geometry of a straight fiber can be defined similar to that of a disk. The radius, *r*, of the fiber and the length, *l*, of the fiber are needed. Similarly the aspect ratio of the fiber is defined as $\mathbf{ar} = \frac{\mathbf{2r}}{\mathbf{l}}$. However fibers, such as CNTs, are rarely ever straight in nature. Even the straightest fiber has some degree of curvature, or waviness to its geometry. In order to model the natural waviness of CNTs a sinusoidal approach was first considered. 3 parametric equations were found to be representative of a sinusoidal wave modeling a CNT [5].

x=acos t y=asint z=bt

The total length, L, is defined to be $t\sqrt{a^2 + b^2}$. To preserve the randomness of a CNT system, the parameter, *a*, was chosen at random for each CNT. The parameter t was randomly chosen between 0 and 4π. Since the total length of the CNT is defined, the parameter b is solved for as $\sqrt{\frac{L}{t}^2 - a^2}$. This approach yielded results only about 40% of the time due to the random choosing of parameters. In order to troubleshoot the problems with this sinusoidal approach not returning good results, t was fixed at π, 2π, 3π and 4π. It was found that 2π produced results more consistently that any other value of t however there was still a 30% chance that the code would fail. It was because of this that other methods were utilized to model the waviness of CNTs. When comparing the images of CNTs in a polymer block, it was determined also that the sinusoidal method was not as realistic as the waviness of a CNT is often more complex than a sine wave can produce. A cubic hermite spline was found to replicate the complex geometry of a CNT by fitting a curve through a set of points defined by a segment length between each point and a degree of tortuosity between each segment [6]. This degree of tortuosity $\theta_{T,max}$ has been used in previous studies with a value ranging from 20° to 40°, in this study 30° was chosen. The number of segments, n, can vary by changing the variable in the code; however the value of 10 segments was used in this study. The length of each segment, L_n, is simply the total length of the CNT divided by n. In order to generate the CNT by this method the function determines the starting location randomly and then chooses a point at a distance of Ln with an angle between the segments, θ_i, chosen randomly between 0 and $\theta_{T,max}$. This is repeated until there are n segments and n + 1 points defining those segments. After the points have been defined, a spline curve is fitted through the points and the output defines the wavy CNT.

CHAPTER III

3.1. Computational Modeling Methodology

The creation of an RVE model is a systematic process of filler generation and data storage. With the parameters defined in Chapter II, MATLAB code was written capable of producing the three different morphologies. Since the different fillers have different parameters, the MATLAB codes have different inputs based on the morphology being modeled. The main computational algorithm handles the systematic creation of these fillers based on the inputs of the user. This algorithm follows a random generation approach to creating the RVE by adding filler geometries until the input volume fraction has been reached at which point the addition of geometries terminates. The figure below shows the high level algorithm for modeling of the RVE models.

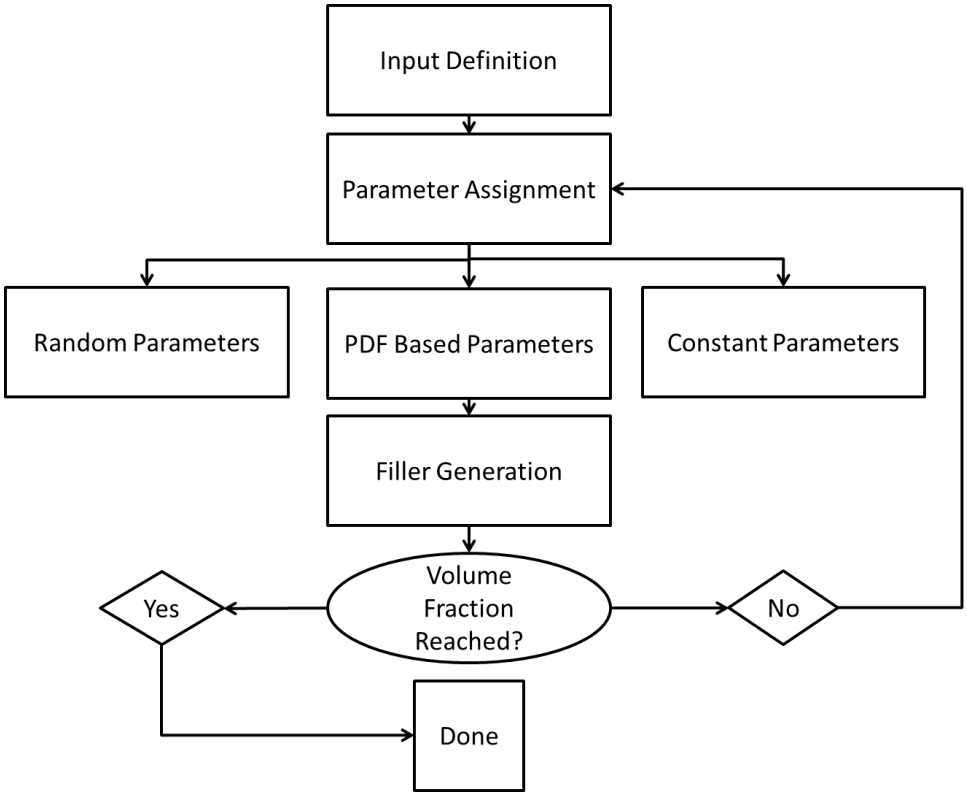

Figure 1: Computational algorithm flowchart

In order to have versatility in the models without having many different codes, the main fiber RVE generation code has 25 inputs. These 25 inputs handle all of the different parameters possible for fiber composites. All of the main inputs are highlighted in the table shown.

Table 1: Selected inputs to the model code

Input Variable (units)	**Description**
Volume Fraction (%)	This sets the max volume fraction for addition of fibers to the RVE.
Interface (nm)	This variable allows for the interface between the fiber and the matrix materials to be modeled as a solid body with different properties
Matrix X Dimension (nm)	The length of the matrix
Matrix Y Dimension	The height of the matrix
Matrix Z Dimension	The depth of the matrix
Length Parameter Flag	This tells the program whether to use constant, random, or PDF based length values
Constant Length (nm)	If constant length is chosen, this is the length of the fibers
Orientation Parameter Flag	This tells the program whether to use no orientation, random orientation, or PDF based orientation values
Waviness Parameter Flag	This tells the program whether to define fibers with no waviness, random waviness, or PDF based waviness.
Diameter Parameter Flag	This tells the program whether to use constant, random, or PDF based diameters
Constant Diameter (nm)	If constant diameter is chosen, this is the diameter of the fibers
PDF Parameters	The use of PDF based values requires that the PDFs be defined. Typical distributions (i.e. Gaussian distributions) can be defined with 2 values.

The main steps for the algorithm are as follows:

1. Generation of a filler particle considering either constant parameters, random parameters, or chose parameters based on a defined input of PDFs

2. Check that the filler resides within the RVE
3. Check that the filler is not penetrating through any other filler particles (extensively discussed in Chapter IV).
4. Update the current volume fraction of the RVE
5. Repeat 1-4 until the desired volume fraction has been met

The algorithm works numerically and thus all of the data generated represent coordinate and dimension data which much be converted and manipulated in order to visualize. The fiber locations are represented by storing 100 points along its axial center, which also includes the effects of orientation, and the diameter. Spherical particles are represented by storing the center of mass and the diameter of the particle. This information can be visualized to a limited extent in MATLAB as shown in the figure below.

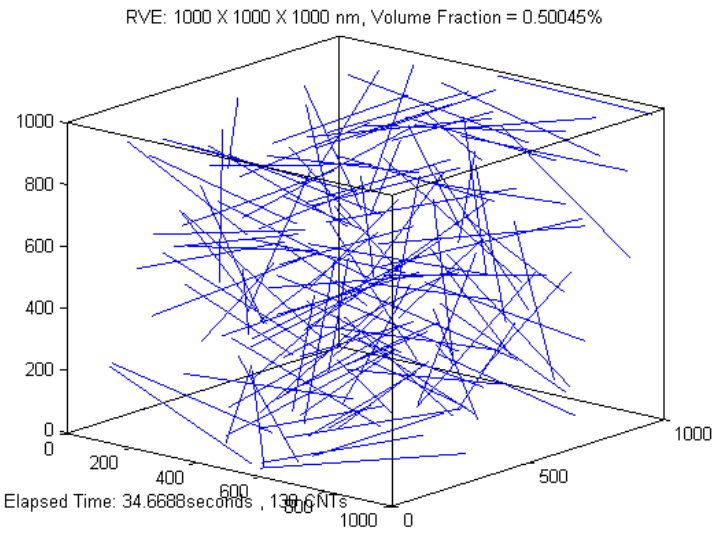

Figure 2: MATLAB visualization of RVE

However visualization in MATLAB offers little use, and thus a 3D modeling software has been integrated into the modeling approach. The 3D model provides much better visualization of the RVE and it provides a simple method by which an FEA analysis can

be performed on the model to computationally estimate material properties of the nanocomposite. The modeling software chosen was Autodesk Inventor because of its well defined Visual Basic (VB) API. Since the number of filler particles could exceed the capabilities of manual modeling of RVEs, the 3D modeling software was required to have a coding interface which allowed for automation of the model generation. The VB API for Inventor provided this. In order to transfer information between MATLAB and Inventor via VB it was found that Excel was a common link between the two. MATLAB writes the coordinate and diameter information along with RVE dimensions to an appropriately formatted Excel document and the VB code in Inventor could be written to parse this document and generate the RVE model accordingly.

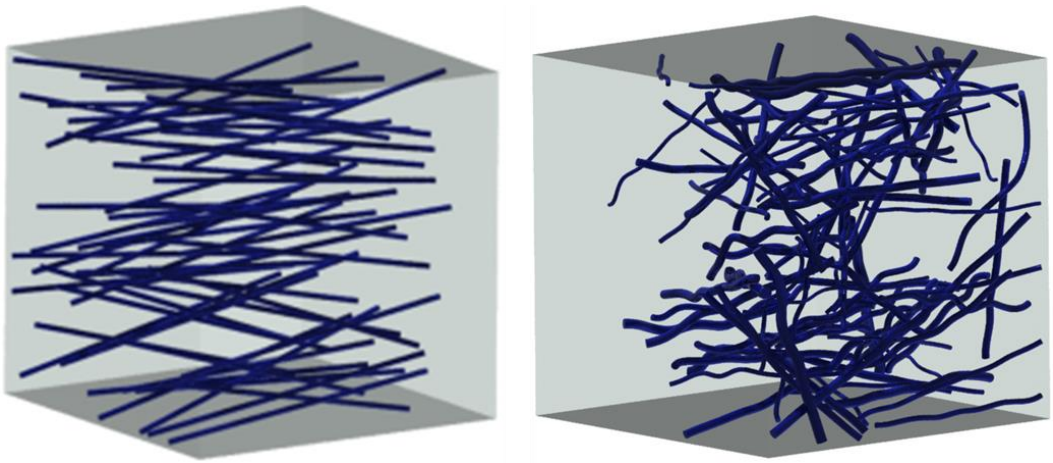

Figure 3: 3D model of sample RVEs

The 3D models could be saved in various formats once generated, however the ACIS SAT file format was chosen because of its compatibility with many different FEA software. The FEA analysis was chosen to be performed in ANSYS due to its well-developed ANSYS parametric design language (APDL) which provided a code interface

which could automate the analysis steps. Using APDL code to generate an initial analysis provided a baseline code which could then be used to create a general code for the analysis of a composite. The generalization was performed in MATLAB itself, where the parameters of the analysis were derived based on the input given by the user. MATLAB is then used to decide how to set up the analysis and then generates the APDL code appropriately. The APDL code is then read as an input to ANSYS and automatically goes through the meshing of the model, the application of boundary conditions and material properties, and the outputting of the results.

In summary, the computational modeling methodology presented is a useful method by which the user specified inputs are used to generate an RVE which matches the input parameters. This RVE is generated first numerically and then imported to a CAD software for 3D representation and then can be imported to FEA software for computational analysis of properties. This framework is the foundation on which it is possible to perform quick ‘what-if’ type analyses to assess how various manufacturing properties and parameters could affect the material properties of a composite. The inputs can easily be changed to represent different scenarios and the differences in the RVE models from these changes should easy to compare and contrast through visualization in 3D or through analysis by FEA.

CHAPTER IV

4.1. Determination of Distance between Non-Straight Fibers

Short-fiber composites cover a broad range of composite systems from metal-matrix composites to polymer nanocomposites. A method has been established by which these composites can be modeled as a filler-matrix system and analyzed in Finite Element Analysis software for mechanical properties. This method assumes that the system is a hard core system, thus the fibers are not allowed to penetrate one another [2, 3, 4]. In this modeling approach, as fibers are added to the representative volume element (RVE) they must be checked for contact or penetration with any other fibers. Checking for this contacting condition between two perfectly straight fibers is a simple process; however the approach taken models the fibers as wavy, cylindrical shapes. This wavy shape is a spine curve and in order to determine the contact distance of two spline curves, the minimum distance between two spline curves is needed. The RVE is in the shape of a block with the cylindrical, wavy fibers inside oriented randomly based on a user-given distribution. The minimum distance between the centers of two fibers possible without penetration is the sum of the radii of the two fibers [1]. This minimum distance is denoted as D_{min}. The spline curve generated by the code represents the axis of the fiber and thus the distance between any two spline curves in the RVE must be greater than or equal to the sum of the diameters assigned to the spline curves along all the points of each spline curve. Three approaches have been considered in order to determine an efficient method by which this shortest distance can be calculated.

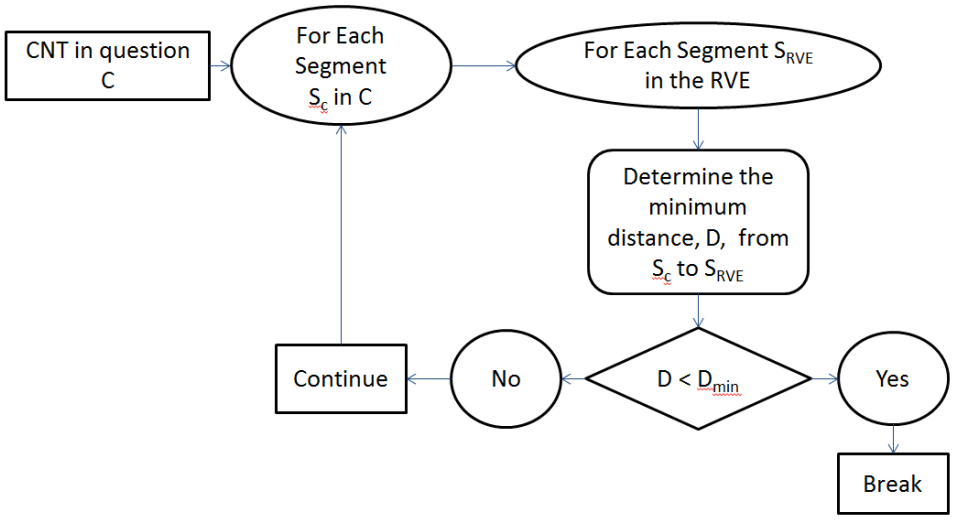

Figure 4: Brute force algorithm

4.1.1. Brute Force Comparison

For the spline used in the modeling, *X* points are defined and the cubic hermite spline curve is fitted to those *X* points. However the spline curve itself is defined by *S* points, generally 10 times greater than *X* or more. These S points in the curve define *S-1* line segments for each spline curve in the RVE. By reducing the geometry from a curve to a piecewise continuous set of line segments and keeping *S* much larger than *X*, if the distance between every segment of the spline in question and every other segment in the RVE is greater than D_{min}, then the two spline will not penetrate each other. This method utilizes the mathematical calculation of the shortest distance between two line segments. Since the time it takes to do this calculation is the basis of the time efficiency of the method, knowing the number of comparisons can estimate the efficiency of this algorithm. The comparison of *S-1* segments with another fiber of *S-1* segments requires $(S-1)^2$ comparisons. Given that there can be *N* fibers in the RVE, there are *N*(S-1)* segments total in the RVE. Thus in order to compare the *S-1* segments of the fiber in question to the *N*(S-1)* total segments in the RVE, $N*(S-1)^2$ comparisons are needed in

all. Since S is fixed and defined, the preceding relation for the number of comparisons gives the algorithm linear time efficiency of $O(n(s-1)^2)$. Yet, even with a linear efficiency as the number of fibers increases the computation time becomes very large. In this particular study, *X* was chosen to be 10 [1] and *S* chosen to be 100, which means an RVE with 50 fibers needs half a million comparisons. The table in Figure 2 shows some results of the number of comparisons and the time taken to perform those calculations. These results signify that a 100 fiber RVE should take around 100 seconds to generate, however these calculations do not take into consideration the process that occurs when a fiber is determined to be penetrating another.

Table 2: Table of computation time required for varying numbers of comparisons.

Number Of Comparisons	Computation Time (s)
1	9.238e-5
9801	.9896
490050	49.985
686070	71.519
980100	99.974

In the event of this happening, since this penetration is unphysical and not allowed for an accurate RVE model, the fiber in question is discarded and a new fiber is generated and compared instead. At low volume fractions this has only a small effect on total computation time as with less fibers there's less chance of penetration occurring. At high volume fractions however, the total computation time increases almost exponentially with time as comparisons have to be repeated until no penetration is detected. Figure 3 shows the computation time of adding *N* fibers to a RVE, with results very different from the ones in Figure 1. While the results are somewhat favorable when considering Table 1, the computation times are much too high when considering Table 2 as at 1.5% volume fraction it takes nearly 1000 seconds. Since short fiber systems manufactured for

electrical properties can have volume fractions up to 10%, this method becomes impractical.

Table 3: Table of computation times for a RVE of dimensions 1680x750x500 with fiber of length 1500 and diameter of 12.

RVE Volume Fraction	Number of Fibers	Computation Time (s)
0.01%	1	.3153
0.15%	6	4.3715
0.50%	19	52.875
1.00%	38	256.578
1.50%	56	881.308

4.1.2. Closest Pair of Points

In order to reduce the number of comparisons and the time needed to create higher volume fraction fillers another method of determining the distance between fibers was considered. An algorithm has been established by which the minimum distance between a set of points can be determined using a recursive divide and conquer approach [5]. Since the RVE can be sliced into cross sections with points in each cross section representing the spline curve intersecting with that cross section, this algorithm was used. This algorithm in itself has an efficiency of *O(nlogn)* which is worse than the linear efficiency of the brute force method, however the goal was that this algorithm wouldn't have to be repeated as much since it is capable of comparing all of the fibers in a given cross section at once. In order to get accurate results from this method, the cross sections taken had to be chosen to be smaller than the diameter of the fiber. From this it was decided that the cross sections would be taken per unit length, so an RVE of length L x W x H, would have L cross sections to analyze giving the algorithm an efficiency of *O(L*nlogn)*. Considering the RVE size used in this study of 1680x750x500, there would be 1680 cross sections to analyze. The algorithm was written such that if no points were detected in the cross section then that cross section would be skipped. In addition if only

1 fiber was detected in a cross section, it would be skipped and if two fibers were detected then the closest pair algorithm was skipped and the shortest distance between the points was just calculated. With these added checks, the 1680 cross sections is only a worse case number and considering such the efficiency is *O(1680*nlogn).* When comparing to the previous algorithm efficiency wise, considering S = 100 and X = 10 with efficiency $O(99^2n)$, the closest pair approach is much more efficient. However in practice this primary approach failed as taking cross sections along the length of the RVE does not account for penetrations that occur in cross sections in the other two directions. The algorithm had to be modified to run along the length, width, and height of the RVE which lowered the efficiency substantially from before. In addition, when considering the number of times the algorithm has to be repeated when a fiber is penetrating a small factor *Z* which decreases the efficiency of the algorithm even more as it has to be run in all 3 directions every time in order to accurately determine the distances. With all of these things considered, trying to actually create an RVE produced the results shown in Figure 4, volume fractions higher than 1% were not tested due to the time needed to compute them. It was evident at this algorithm was more efficient than the brute force, however this is only true in very ideal cases, specifically when the volume fraction is very low. In practice however, this method too was impractical for realistic modeling as the computation time becomes extremely high with increases in volume fraction.

Table 4: Computation time to create an RVE with the Closest Pair algorithm.

RVE Volume Fraction	Number of Fibers	Computation Time (s)
.01%	1	0.315303
.1%	4	1.80544
1%	38	636.774596

4.1.3. Selective Comparing

Due to the brute force method being more efficient at higher volume fractions than the closest pair algorithm, it was revisited in an attempt to make it more efficient. From the set of segments being compared, it was found that much time was wasted comparing segments which could never be penetrating. The algorithm could be revised to only consider segments which lie close enough to be penetrating. By determining the maximum and minimum values in each direction of the fiber in question, a boundary can be drawn such that if segments do not have at least one endpoint within the boundary then it is not considered. Since D_{min} is defined, the boundary is defined by adding D_{min} to the maximum and minimum values in the x, y and z direction of the fiber in question. This selective comparison drastically reduces the number of comparisons needed to determine if a fiber is penetrating or not, which in turn drastically reduces the time needed to perform the modeling of an RVE, especially at higher volume fractions. The table in Figure 5 shows some of the results from this selective method.

Table 5: RVE modeling times when using selective comparing.

RVE Volume Fraction	Number of Fibers	Computation Time (s)
1.00%	38	17.851
1.50%	56	36.235
2.00%	75	117.980
2.50%	93	228.318

As can be seen in the table above, the time for 1.00% volume fraction has decreased an order of magnitude from both prior methods. In addition the code can generate RVE in excess of 2.50% now in much less time. Figure 6 shows a comparison of the three approaches, illustrating the profound change that the selective comparing has on the running time of the algorithm.

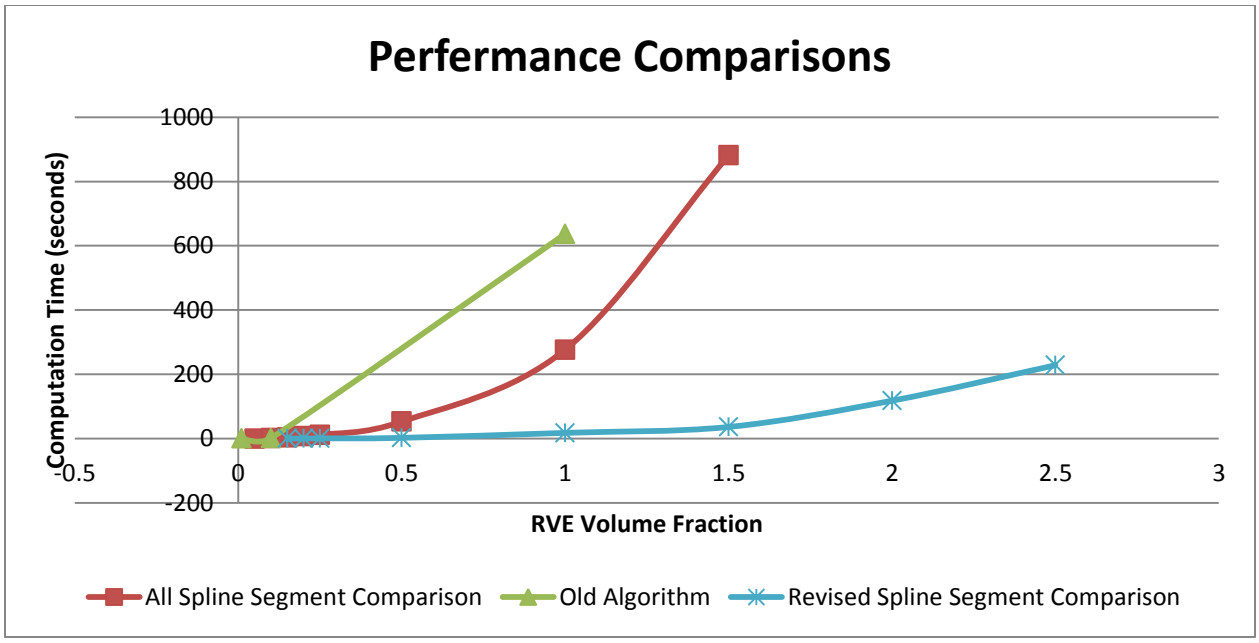

Figure 5: A comparison of the computation of the three preceding methods.

4.1.4. Parallelization of the Algorithm

While the results obtained from the selective comparison approach are the most time efficient, the time needed for higher volume fractions needs to be reduced even further. The flowchart in Figure 7 shows an overview of the entire computational algorithm for modeling a fiber composite system. In order to achieve this speedup, the algorithm was parallelized. The only part of the algorithm which is non-sequential is the code that runs within the contact and penetration check block. This part of the algorithm does the most work and takes the most time, even after the modifications applied in the selective comparison section. Figure 1 outlines the process by which the comparisons take place; the only difference being that rather than S_{RVE} representing all the segments in the RVE, S_{RVE} only represents those segments within the boundary region stated in the last section. From this outline it can be seen that all of the comparisons could be parallelized with no penalty to accuracy as no information is needed for a comparison but the location of one

segment on the fiber and the location of one segment inside the RVE. This entire algorithm is written and implemented in Matlab and in order to perform the parallelization, the Parallel Computing Toolbox [7] was used. The loop structure in Figure 1 is accomplished by a for loop, using the Parallel Computing Toolbox this can be modified in to a parallel for loop which distributes the work of the contents of the loop amongst Matlab 'workers'. These workers can be either multiple cores on a computer or multiple nodes in a cluster.

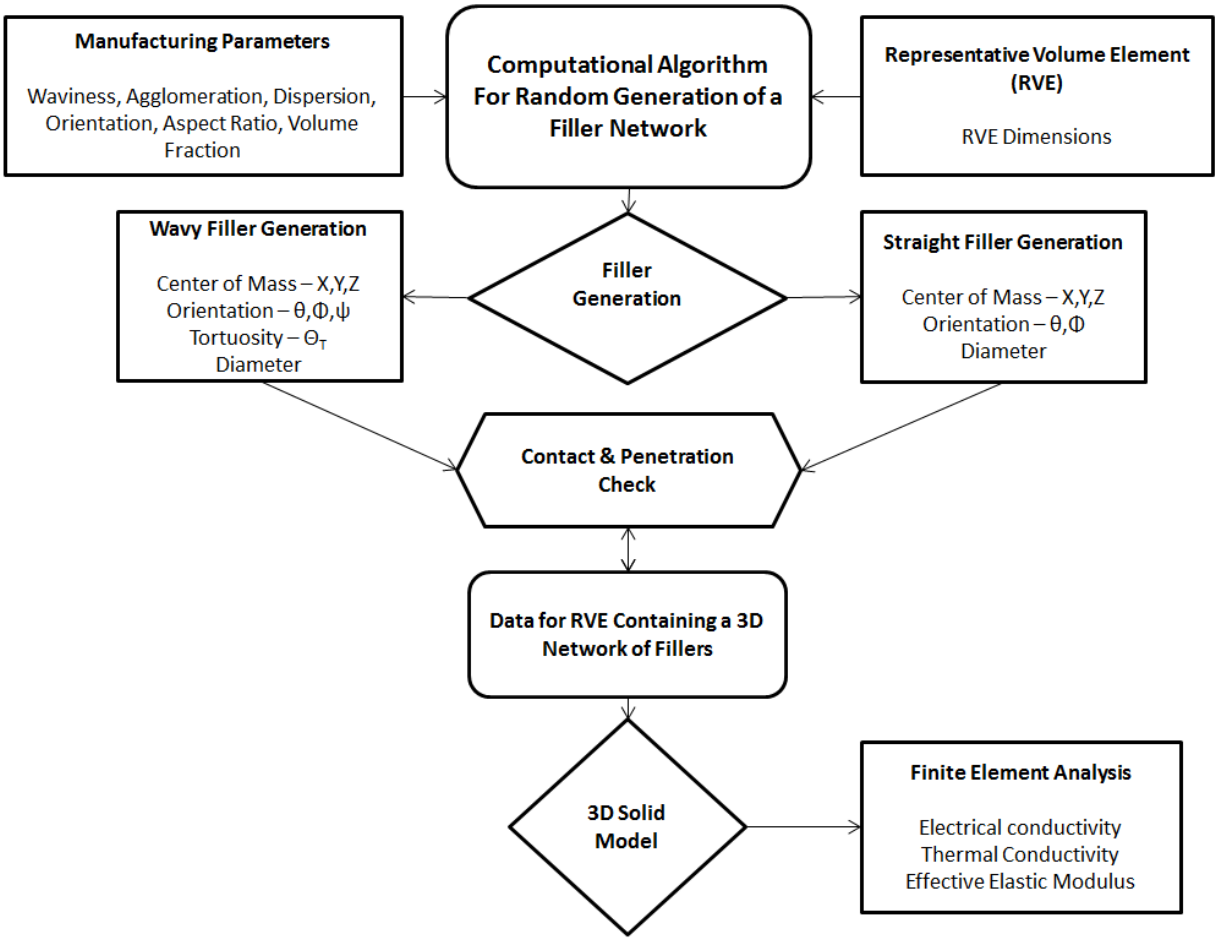

Figure 6: This flowchart represents the full algorithm which generates a fiber composite RVE.

Due to the structure of the parallel for loop, in Matlab called the *parfor*, the code had to be modified to be compatible with the *parfor* loop. In the process shown in Figure 1, the

distance was checked against D_{min} after every segment was compared and the Yes/No in the figure correspond to a True/False value for penetration. If this value is True then a penetration has been detected and the code needs to break. For a parallel for loop implementing this, the segment comparisons would happen in parallel and this value was assumed to be the same, holding a Yes or No depending on the results of the check and breaking out of the loop when a Yes or True was detected. However, the break statement cannot be used in Matlab and thus the structure of the loop was modified to look like what is shown in Figure 8. In order to obtain the optimal loop structure, each segment distance D and each D_{min} are calculated and added to a combined vector. The combined vector of D and D_{min} give a simple way to determine penetrations as $D\text{-}D_{min} \geq 0$ for all non-penetrating segments. The only setback of this approach is rather than breaking out of the loop once a penetration is detected, the code has to finish the current loop prior to determining that it needs to break. However this setback is compensated for as the parallel algorithm runs fast enough that comparing all segments in the loop before determining it needs to break has minimal effect on running time. In addition to the loop structure itself, the syntax for the parallel loop was modified from the original segment check code in order to reduce communication overhead in Matlab. These variables contain smaller sections of the main input variable so that Matlab doesn't have to search for these specific sections.

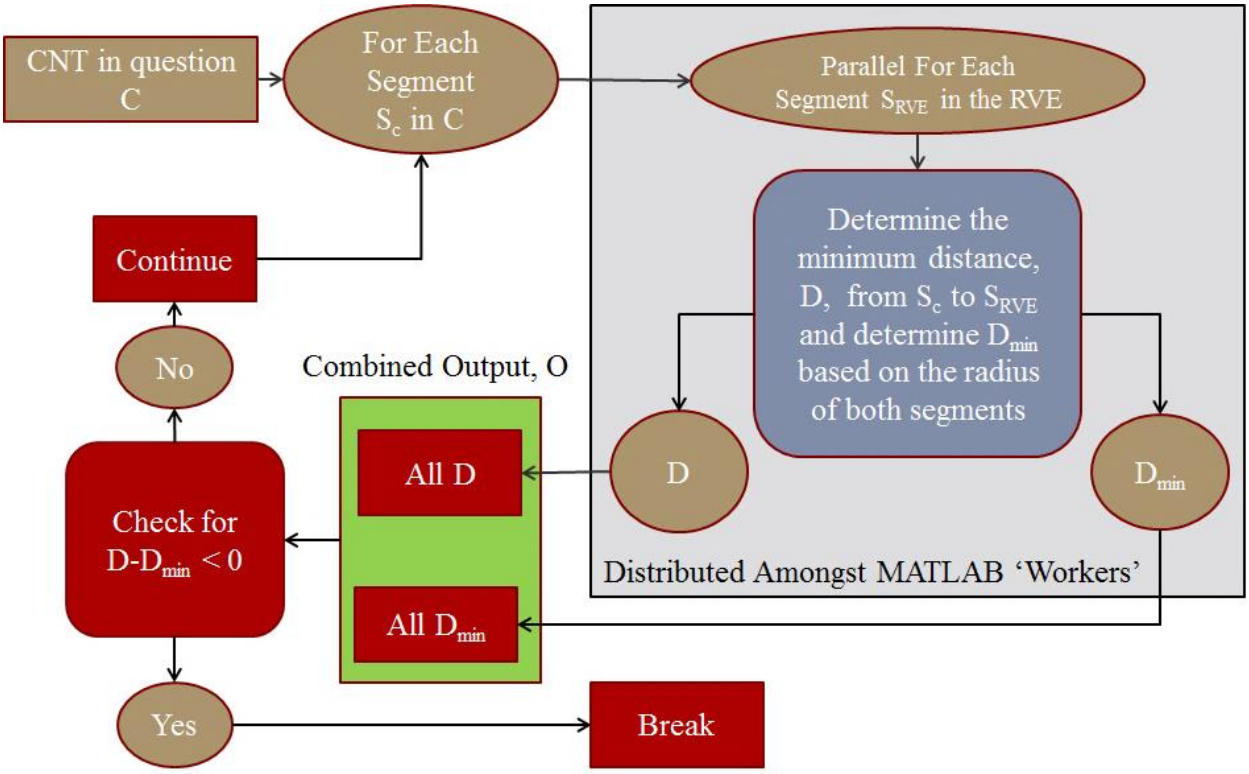

Figure 7: The loop structure for use in a parallel loop

4.1.5. Results and Discussion

In order to quantify the effect of parallelization on the code running time, it was run on various computing resources to see generalized trends as the number of comparisons increases. Figure 9 shows the results of a preliminary test ran on a computer with an AMD Phenom II 6-core processor @ 3.2GHz. From these results there can be seen an almost constant 8x speed up for the parallel algorithm over the regular algorithm for a number of comparisons greater than 2000. However, under 2000 comparisons the trend was surprisingly different and Figure 10 shows a closer view of this region. In Figure 10, it can be seen that until the number of comparisons is approximately, the parallel algorithm actually performs worse than the regular one. Because this test was the

preliminary test and the machine tested on was in use, it was thought that this region was a mistake and that other tests would correct this. However

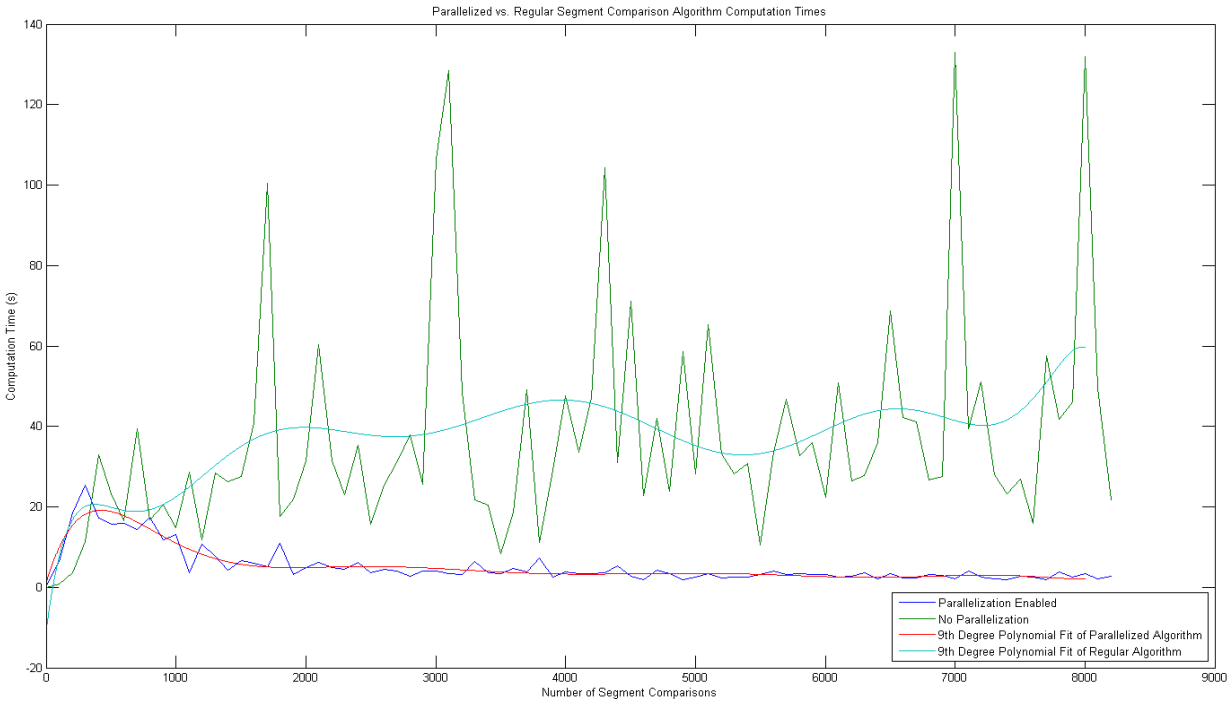

Figure 8: Computation time comparison of parallel and regular algorithm

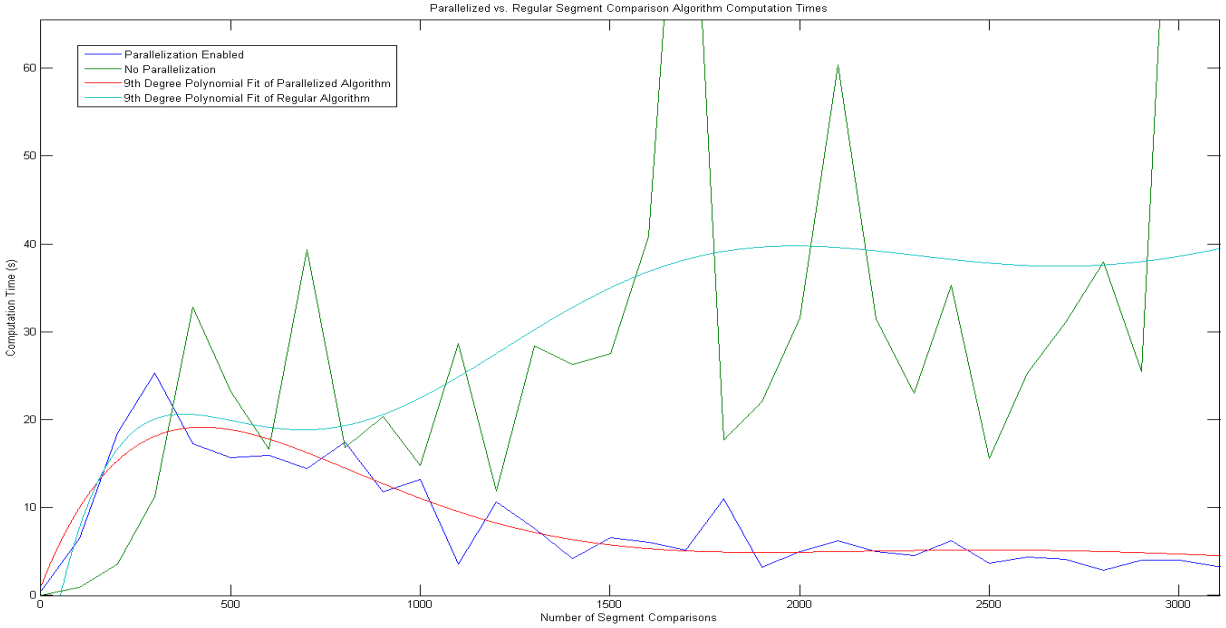

Figure 9: Close up of the region below 2000 comparisons

This was not found to be the case as further testing confirmed what could be seen in Figures 9 & 10. There is a region, though small, in which the parallel algorithm isn't faster than the regular one. In order to further test the FoRCE HPC cluster was used, of which have nodes with 6-core AMD Opteron processors @ 2.4GHz. The graph in Figure 11 shows the fits of Computation Time vs. Number of Comparisons up to 15000 comparisons on the FoRCE [6] cluster. When looking at the fits of this data for the cluster it is first apparent that the intersection of where the parallel algorithm actually becomes more efficient is at a higher number of comparisons than in Figure 9. Using 6 cores and 8 cores, both don't perform better than the regular algorithm until approximately 1000-1200 comparisons are needed. This threshold value depends on the number of cores used. In addition, these fits imply that the higher number of cores used in the parallel loop, the higher the increase in inefficiency of the parallel algorithm below the threshold value. While this has considerable implications on the code since at low volume fractions there are few segments to compare, it is hard to predict the value at which the threshold occurs for different machines. However it is clear that the parallel algorithm should be used only above this threshold. From Figure 9 and 11 it can be seen that the speedup is basically independent of the number of cores at this number of comparisons and thus an increased number of comparisons was needed to see the effect of more cores on the performance of the parallel algorithm. Figure 12 shows the results of the data collected from 15000 comparisons to 1.5 million comparisons. Primarily, a trend can now be seen clearly between the number of cores and the speedup attained from parallelization. In this figure the plot for the regular algorithm is not shown as the

computation time required to obtain 1.5 million or even half a million, comparisons was much greater than either the 6 core or 8 core parallel runs.

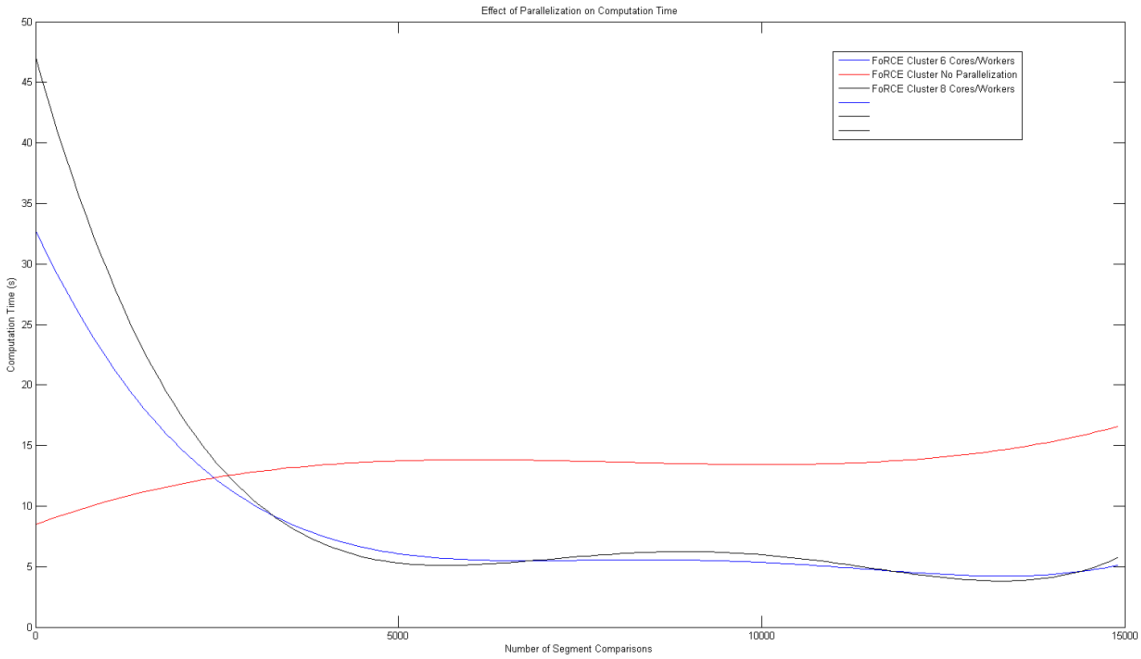

Figure 10: FoRCE HPC cluster results up to 15000 comparisons

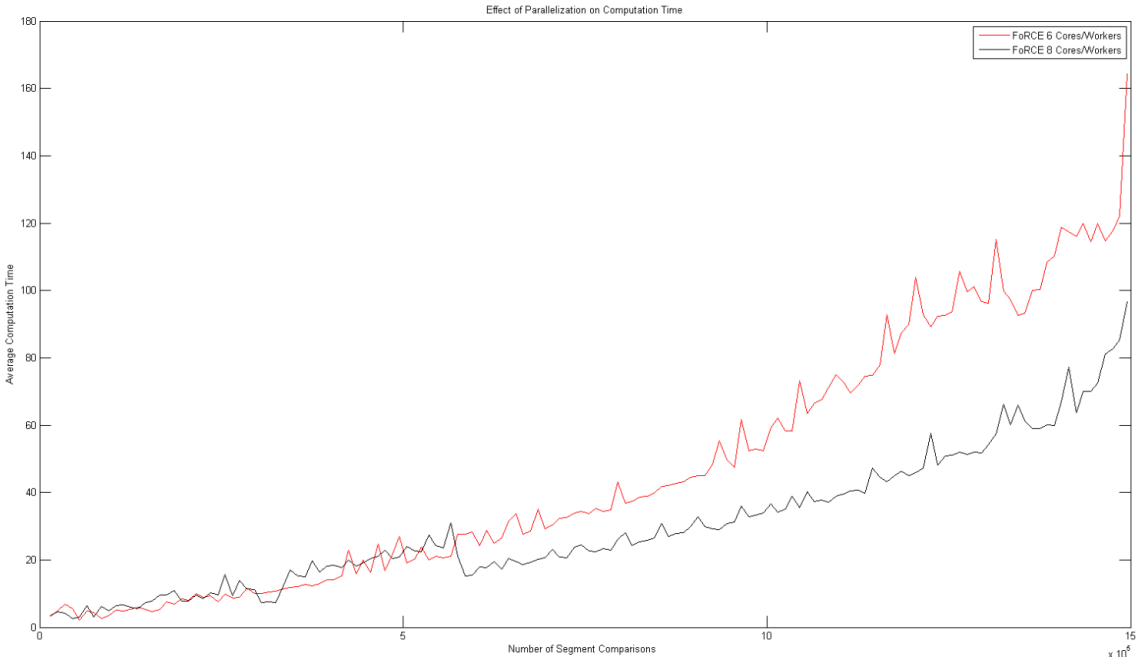

Figure 11: Performance of the parallel algorithm at high numbers of comparisons

From this plot and the previous, it can be concluded that the computation time improvement per number of cores above 6 is generally equal until approximately 600,000 comparisons. Along with the threshold value for comparisons needed for the parallel algorithm to be useful, this number can be useful in determining how many processors to reserve for a set of comparisons. Since efficiency is important, knowing these two threshold values can increase performance if the number of comparisons is less than the lower threshold and maximize the work done per processor for comparisons less than the higher threshold. It can be concluded also by looking at these plots that the effect of parallelization is parabolic which is why these thresholds are necessary. If the performance of the regular algorithm can be said to be linear then the parabolic performance of the parallel algorithm will intersect the linear line at some low value, which is the lower threshold. While this work has improved the running time of the code substantially future work is needed to further optimize the code based on the number of comparisons and the resources available to Matlab.

CHAPTER V

5.1. Image Analysis Algorithm

The method developed to determine the orientation of a fiber from a cross section SEM is composed of 6 steps essentially which are repeated for each fiber cross section detected. The flowchart below outlines the process.

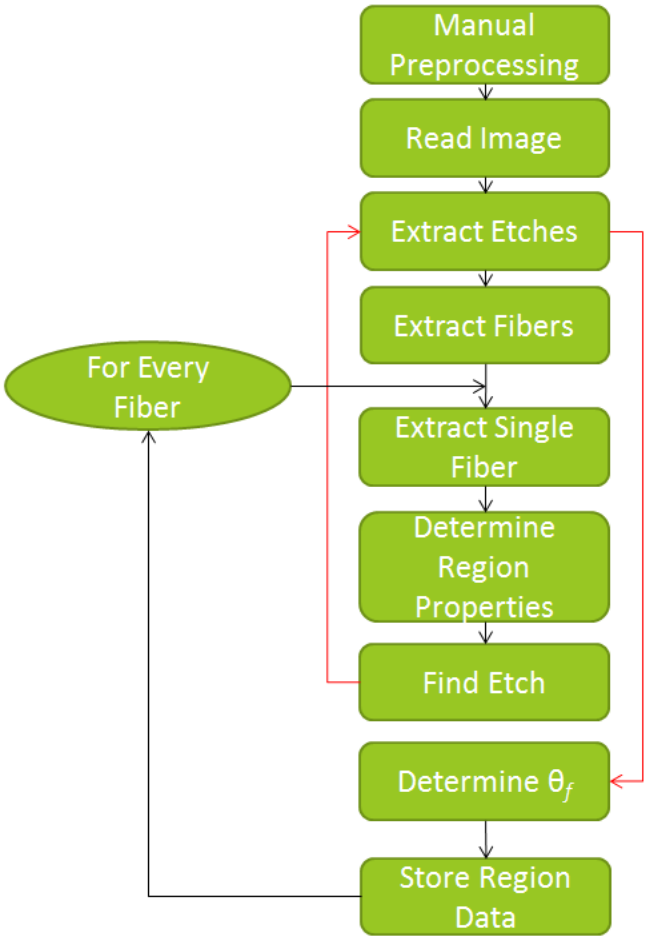

Figure 12: Image analysis procedure

The primary step is to check to see if manual preprocessing of the image is necessary. Manual preprocessing is required for the image if there are sections where fiber cross sections are overlapping because the code cannot differentiate between the two fiber

areas easily. An example of the manual preprocessing is shown in Figure 1 where two black lines are added to the image to separate two overlapping fibers cross section areas. The image is then read into MATLAB it is converted from an RGB image into a grayscale image. From the grayscale image, the location of the etch marks are determined with a binary conversion factor. Since the etch marks are very dark on the image, it is relatively easy to separate them with a binary image conversion. The image containing the extracted etches is then stored. Figure 1 shows the original image and Figure 2 shows etches obtained from the image after binary conversion.

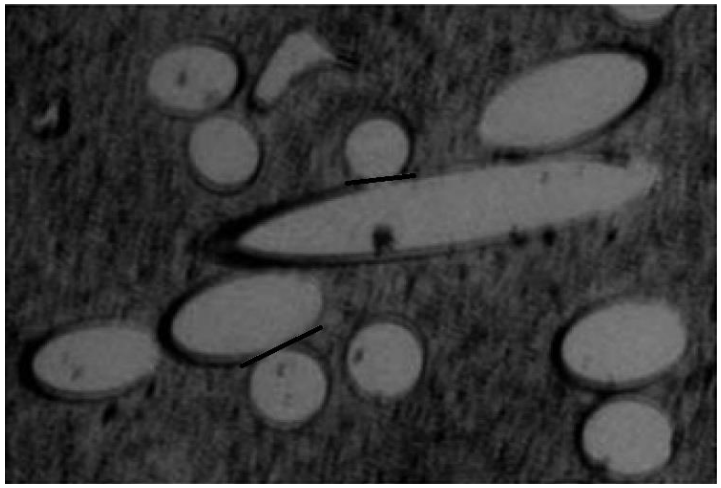

Figure 13: Manual Preprocessing

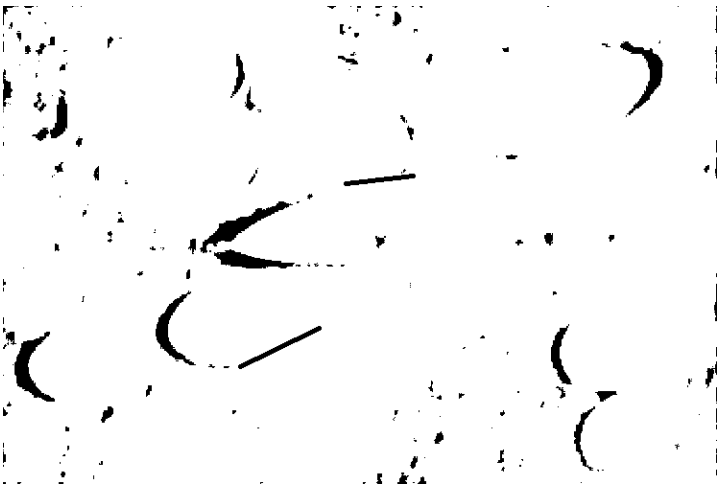

Figure 14: The etch locations obtained from the original image

Using another binary conversion, the fiber cross sections are then extracted from the original image. The contrast between the relatively white color of the fibers makes it easy to extract them from the original image. Figure 3 shows the obtained fiber cross sections from the original image.

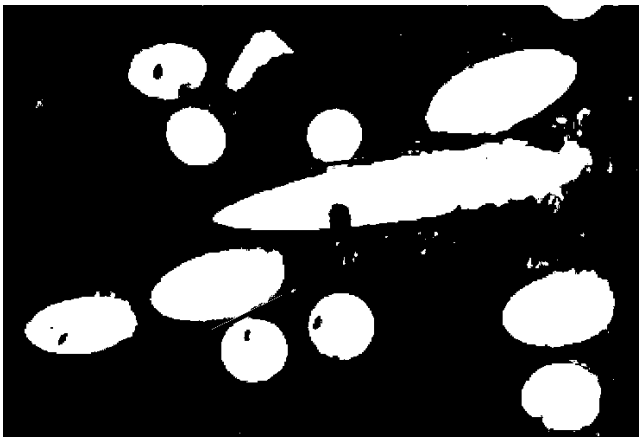

Figure 15: Fiber cross sections obtained

In order to be able to obtain the properties for a single fiber cross section a method was developed to find only the location of the cross section boundaries and traverse them. Since each fiber boundary is only 1 pixel thick, each white boundary pixel is surrounded by exactly 2 other white pixels. Thus traversing a boundary implies that the first pixel traversed will also be the last. By stepping in one direction for each pixel in the boundary until reaching the original pixel, the outline of one fiber is determined. In order to obtain the fiber boundaries, morphological operations were used in MATLAB. First the image is refined using 'majority' and 'fill' commands which help to reduce noise and fill small areas with the appropriate pixels. Finally the 'remove' command removes all white pixels except those located at the boundary. Figure 4 shows the image after these operations.

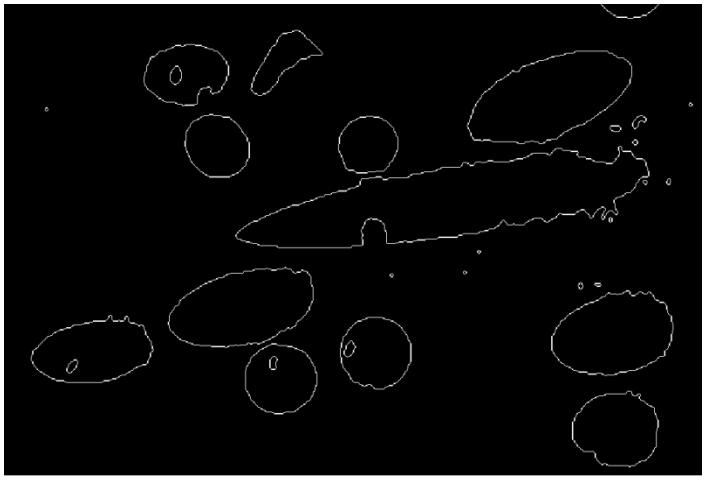

Figure 16: Fibers after processing

From this image a white pixel location is chosen at random and the boundary of the fiber to which that pixel belongs is traversed. During this traversal each pixel location visited is stored in a separate image of equal size to the original image which essentially makes a copy of the fiber boundary. In this copied image the boundary is then filled and the region properties of the fiber are determined. Using the *regionprops* command in MATLAB the (x,y) location of the centroid, the major and minor axes, the area and the orientation angle α can be determined. The orientation angle is calculated as follows: $\alpha = \sin^{-1} b/a$. The in plane orientation angle is determined by MATLAB as well as shown in Figure 5. The in-plane orientation angle is ambiguous, however, as determined because the fiber could be oriented at the given angle or at π plus the given angle. By considering the location of etching, this ambiguity can be removed. The code finds the leftmost and rightmost locations on the boundary of the fiber and then steps 10 pixels in those directions. For example the location at 10 to the left of the leftmost pixel in the fiber boundary image is compared with the same location in the etch image. If the etch image is black in the location in the etch image then the in-plane orientation is as calculated in MATLAB. If the pixels are not black in this location then either the etch

mark is on the right side or there is no etch and the orientation obtained in MATLAB is then increased by π. Figure 6 shows this process and Figure 7 shows the explanation for the orientation as explained in Vélez-García's paper [14].

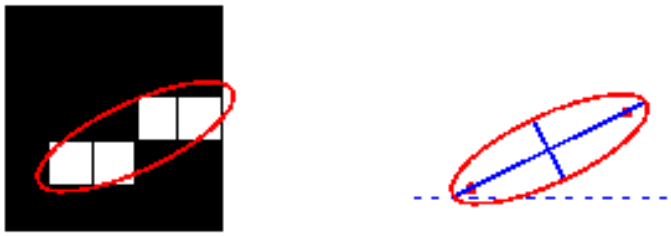

Figure 17: Calculation of in plane orientation

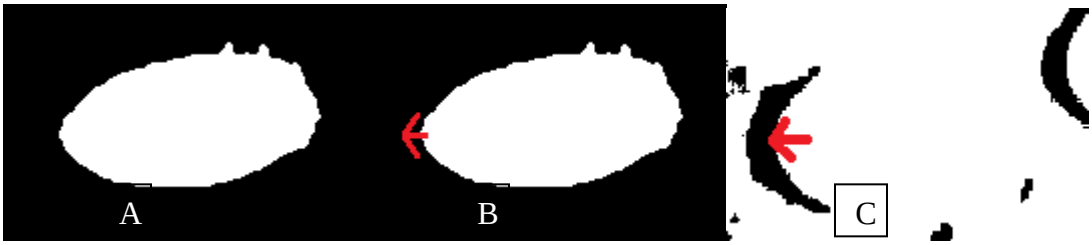

Figure 18: Etch location identification

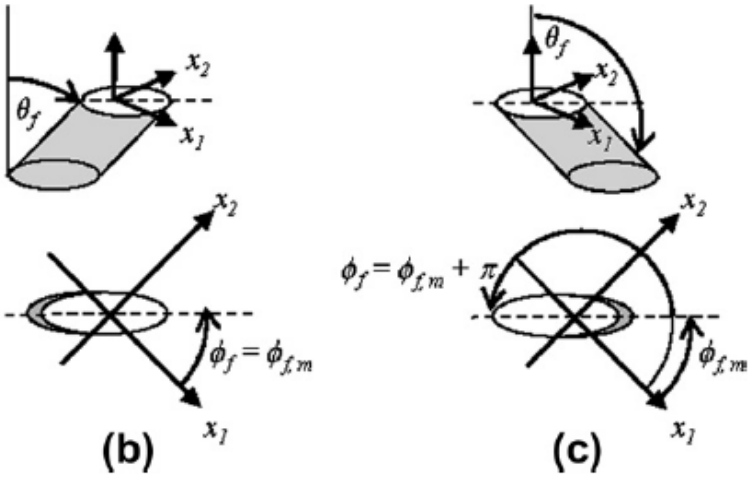

Figure 19: In plane orientation explanation

Once all information has been determined, the data is stored and the original image is given color based on the etch location. If the etch was found on the left side then the

fiber will be colored red. If the etch was not found on the left side, it is colored blue. An example output image is shown in Figure 8.

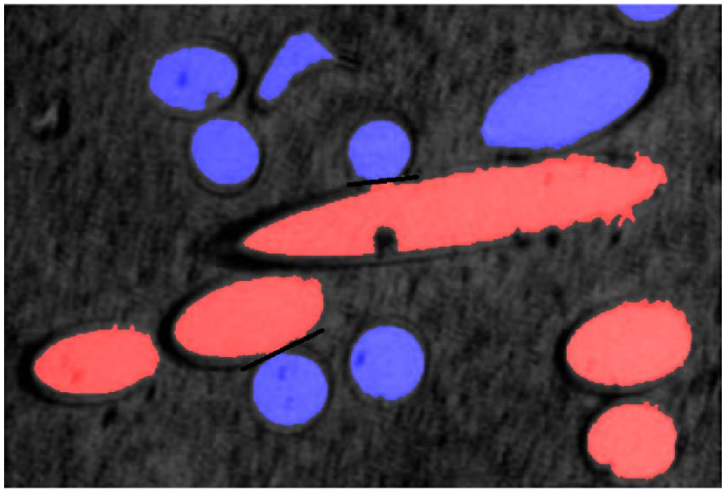

Figure 20: Example Output Image

5.2. Image Analysis Approach for Length & Waviness

A code has been developed to extract network information from 3-D collagen gels which was demonstrated with the capability to provide insight on the lengths of fibers through image analysis of confocal images. This code was modified to work with SEM images of so that the length distribution, and waviness distribution (which is based on length), could be determined. The Fiber Extraction code (FIRE) developed works on a principal of nucleation and local maxima points. Using distance transforms and parameters set by the user, the program finds nucleation points and maps the fiber network through the local maxima points. The code was modified to output both the length along the fiber and the endpoint to endpoint length of the fibers. The figure below shows the typical process by which the fiber network is obtained through FIRE.

(Since the FIRE code has pretty extensive documentation and a paper we can reference and because I don't exactly know how all parts of the code function, I didn't explain too much into how the code works.)

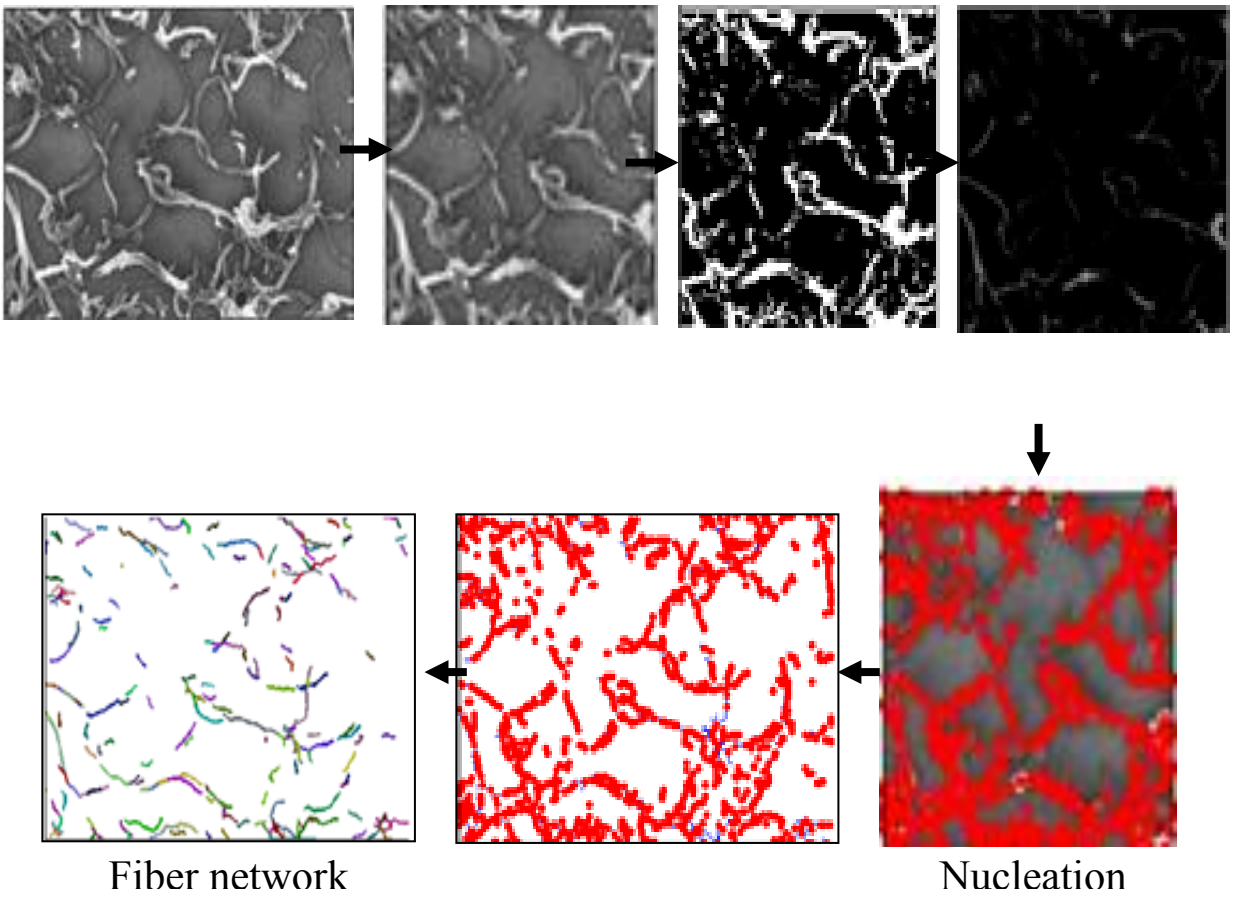

Figure 21: Typical process steps for FIRE

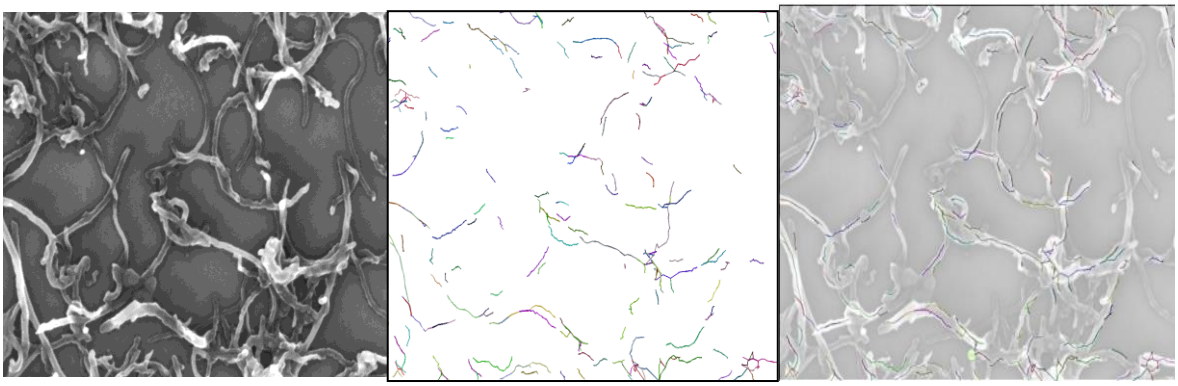

Figure 22: Left: Original Image, Center: Fiber Network after FIRE, Right: Fiber Network superimposed on the Original image

5.3. Image Analysis Approach for Diameter

In order to obtain the diameter distribution, two methods were considered. Building on the method described for the orientation distribution above, the diameter could be calculated with the major and minor axes, *a* and *b* respectively. For a perfect circular cross section, $a = b$ which represents the diameter. For values of *a* and *b* that are fairly close, the diameter is taken as the average of the two. However if the value of *a* is larger than *2*b* then the diameter is taken as *b*. The other method considered built upon the FIRE code used for the length and waviness distribution. Since the diameter along the length of the fiber may change, the FIRE code was modified to return the average radius along each fiber detected. Since the distance transform, on which FIRE relies heavily, calculates highest distance at the middle of the fiber, this distance is representative of the radius.

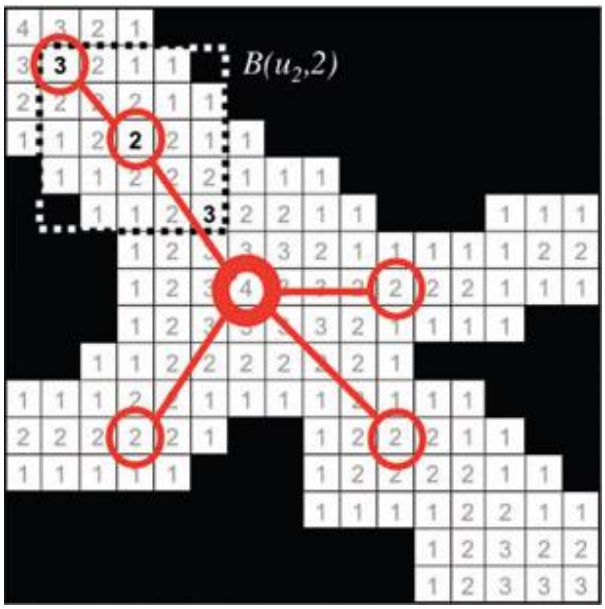

Figure 23: Example values returned by the distance transform

5.4. Probability Distribution Functions from Image Analysis

The purpose of the image analysis algorithm is to derive the parameters for a Probability Distribution Functions (PDF) which fits to the data. These numerical distributions are based on normal, lognormal, extreme-value, gamma or beta distributions and are implemented in the RVE model to create distributions of RVE filler parameters which match the distributions found in an experimental composite. Sections 5.2 and 5.3 discuss the different parameters for which data can be extracted from an image. Orientation information is extracted as two values, in plane orientation from -180 degrees to 180 degrees and out of plane orientation from 0 degrees to 90 degrees. The diameters and lengths are extracted in terms of pixels, however using a simple conversion based on the scale of the SEM or TEM image the pixel dimensions can be converted to nanometers. The waviness distribution is derived from the total length and the endpoint to endpoint length of each fiber. This relationship is defined as the straightness parameter, $P_s = \frac{L_{ep}}{L_f}$, where L_{ep} is the endpoint to endpoint length and L_f is the total length. The value ranges from approximately 0 to 1, with 1 representing a completely straight fiber. Waviness increases with decreasing P_s. The dispersion of the fillers can be extracted also by recording the center of mass locations for each filler and converting the pixel distances to nanometer values. Using MATLAB, the data extracted can be plotted as a histogram which visualizes the general shape of the distribution of the data. Different PDF types are selected as suggested by the different distributions of data viewed in the histogram. The figure below shows a sample distribution of the out of plane orientation Phi. It can be seen that this distribution is very close to that of a normal Gaussian distribution with its

mean around 45 degrees. As such the red line represents the selected normal distribution PDF fit to the data obtained from the image analysis.

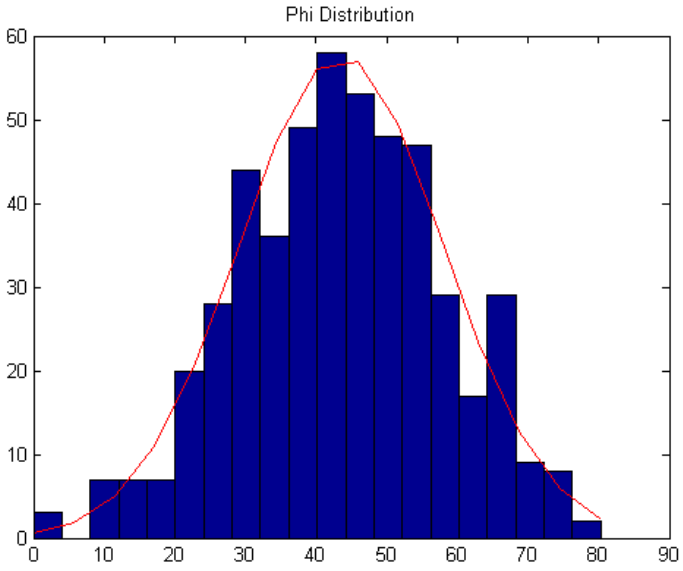

Figure 24: Distribution of Phi

To demonstrate the necessity for different PDFs based on the distribution of data, the figure below shows a lognormal distribution for the data extracted on the diameters of the fibers.

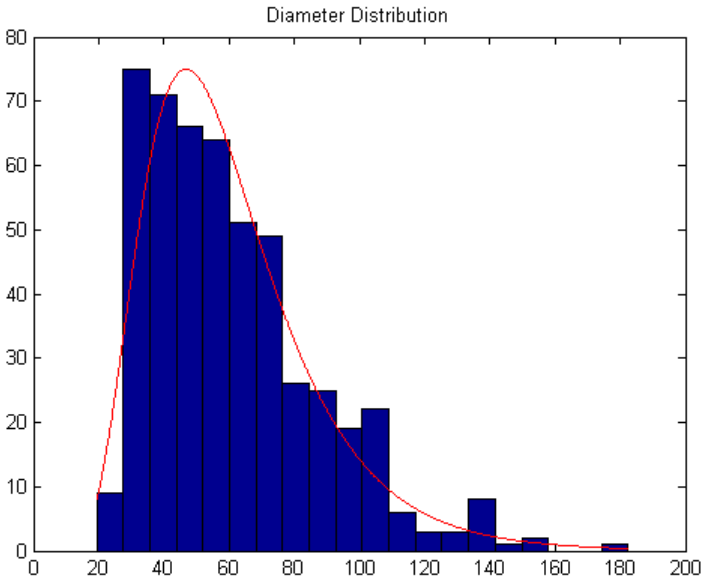

Figure 25: Distribution of diameter

The appropriate fitting of the PDF to the data extracted from image analysis is used to find the two parameters which define the PDF itself. These two parameters are then used in the RVE model code as inputs which will define the diameter, length, orientation, and/or waviness based solely on a random number generated from the two parameter defined PDF for each respective property. The centroid distribution which represents dispersion in the RVE has not been implemented yet into the model because no general correlation between the data can be found. The current model assumes that distribution is random, that is filler center of mass locations are chosen at random using the MATLAB random function which is based on a uniform distribution.

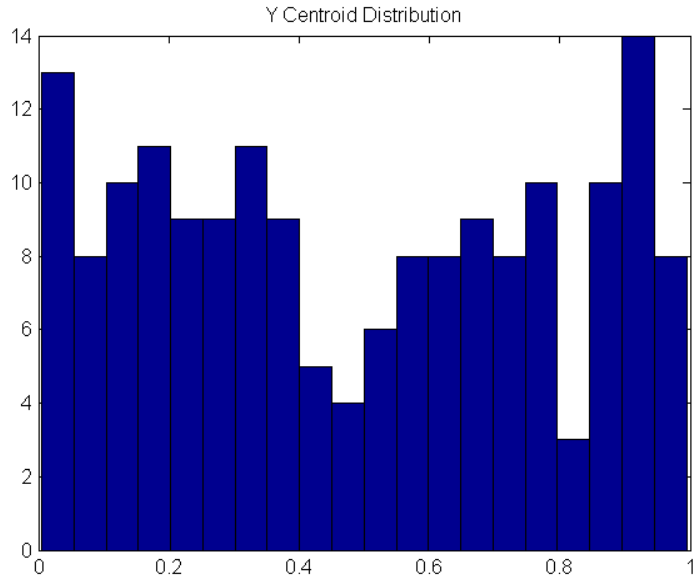

Figure 26: Distribution of centroid locations in the Y direction

The figure above shows the distribution of the centroids, as a ratio of total image size, and it can be seen that no general distribution could be fit to this data. Furthermore, the data seems to imply that a random uniform number could be sufficient to model fiber nanocomposites accurately. However, it is known that the dispersion of fibers is a very important factor in the composite material properties, thus more work must be done to

determine the true distributions possible for center of mass locations in fiber nanocomposites. While the data obtained thus far implies there is no PDF which determines dispersion, it also implies that the contributing factors for the performance of the composite are not reliant upon the dispersion. Image analysis of more samples of fiber nanocomposites will be helpful in obtaining more clarity on the dispersion of the fibers and how this dispersion can be fully quantified using a PDF.

CHAPTER VI

6.1. Integration and Automation of Programs

The complexity of the modeling approach presented lies in the multiple programs required in order to use the models. Due to this, a preliminary integration framework has been developed and tested for use with the RVE model codes. The integration was done using Visual Basic for Applications (VBA). VBA is a good candidate for integration of various different applications because it is written specifically for communication between different Microsoft Windows programs. Furthermore the Visual Basic (VB) language is relatively simple to learn with easy to integrate graphics. Thus a graphic user interface (GUI) was developed which used VBA in the background to call the various programs necessary for the RVE modeling code. This interfaces all of the programs, MATLAB, Excel, Inventor, and ANSYS, and parses the user's inputs appropriately. The specific case for which the GUI was developed was the analysis of fiber nanocomposites for their stress-strain properties. As such, ANSYS was chosen as the FEA software for the analysis due to its ADPL coding capabilities.

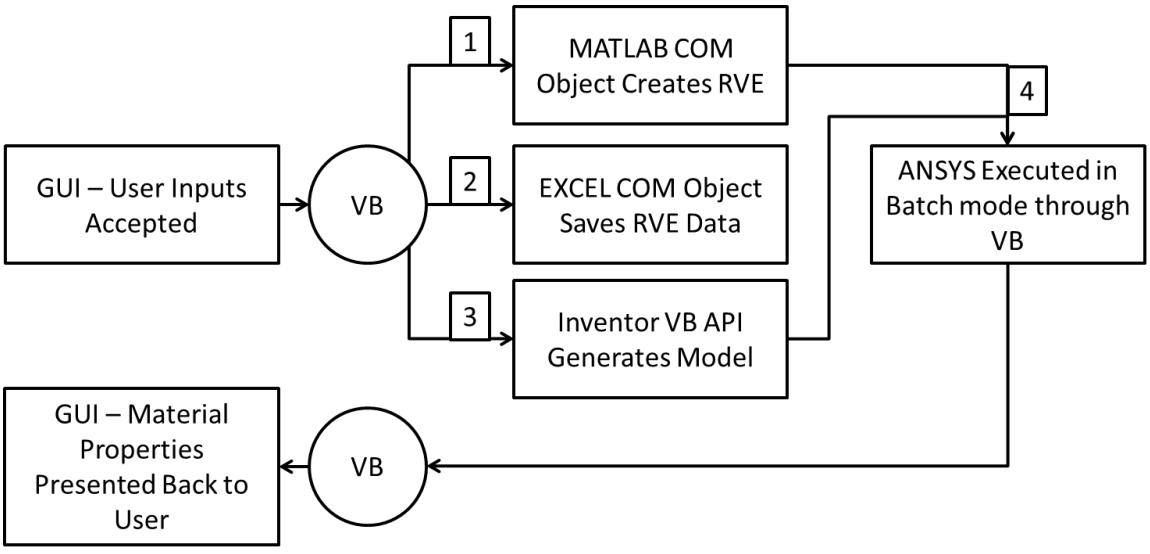

Figure 27: GUI operations

The GUI first page gives the users the options of performing Mechanical, Electrical, or Thermal analyses, as all are capable with the RVE models that have been developed. At time of writing, however, only the mechanical analysis provides meaningful results. The next page presents the options of performing compressive, tensile, or shear loading on the RVE modeled. Since the GUI automates the entire process, the loadings are all defined within the APDL code which is generated by the RVE modeling code in MATLAB. The final page presents the user with the option to use fiber or spherical particles as the filler geometries.

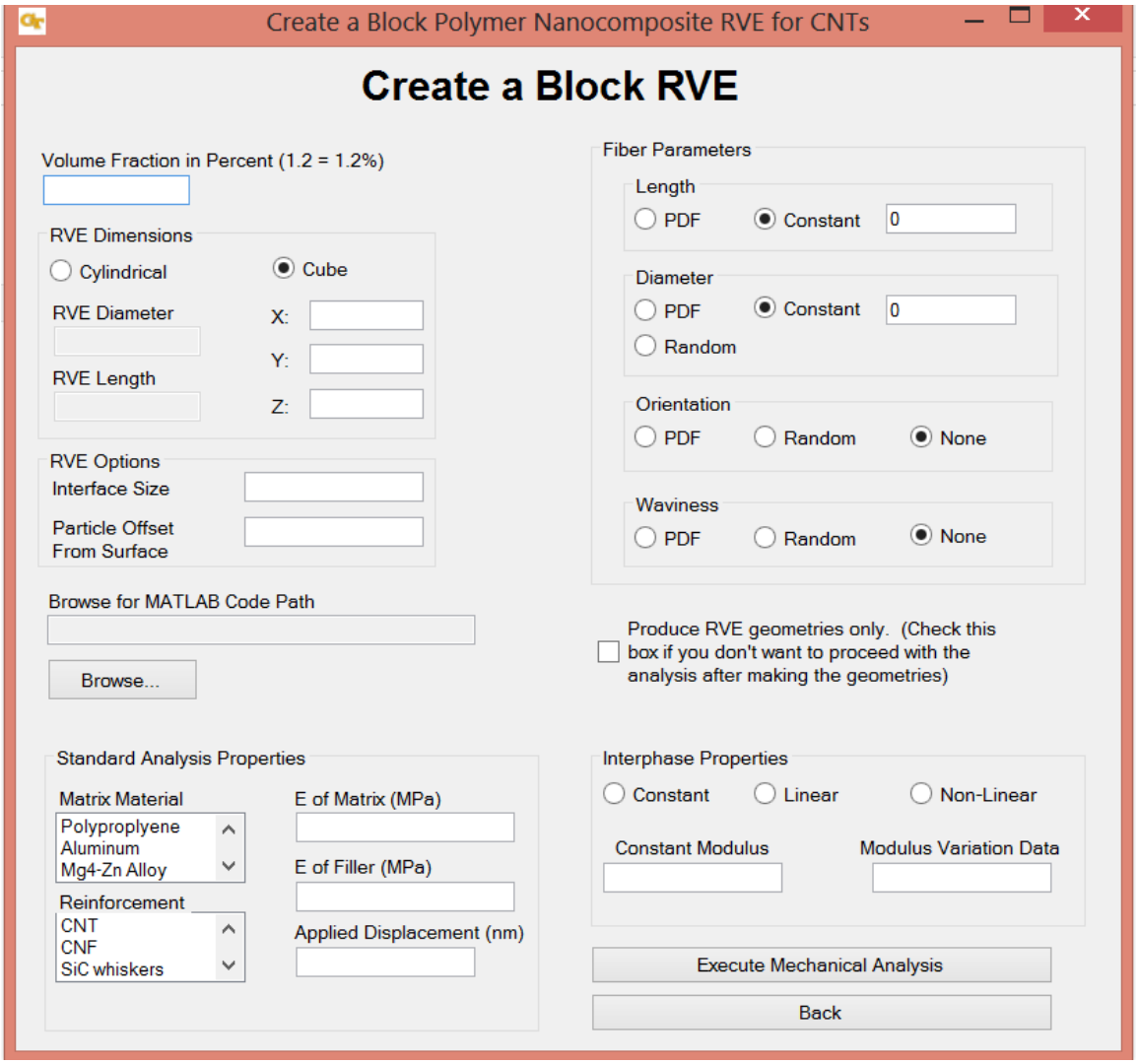

Figure 28: Fiber nanocomposite GUI screen

Once these high level properties have been selected, the user is presented with a page which allows them to input the material specific properties and inputs discussed in Chapter III. The figure above shows the screen presented to the users for the input of the values required. In addition, the PDF parameters are defined on a separate screen which presents itself if a PDF is chosen by the user. Five typical distribution functions are allowed, normal, lognormal, extreme-value, beta, and gamma.

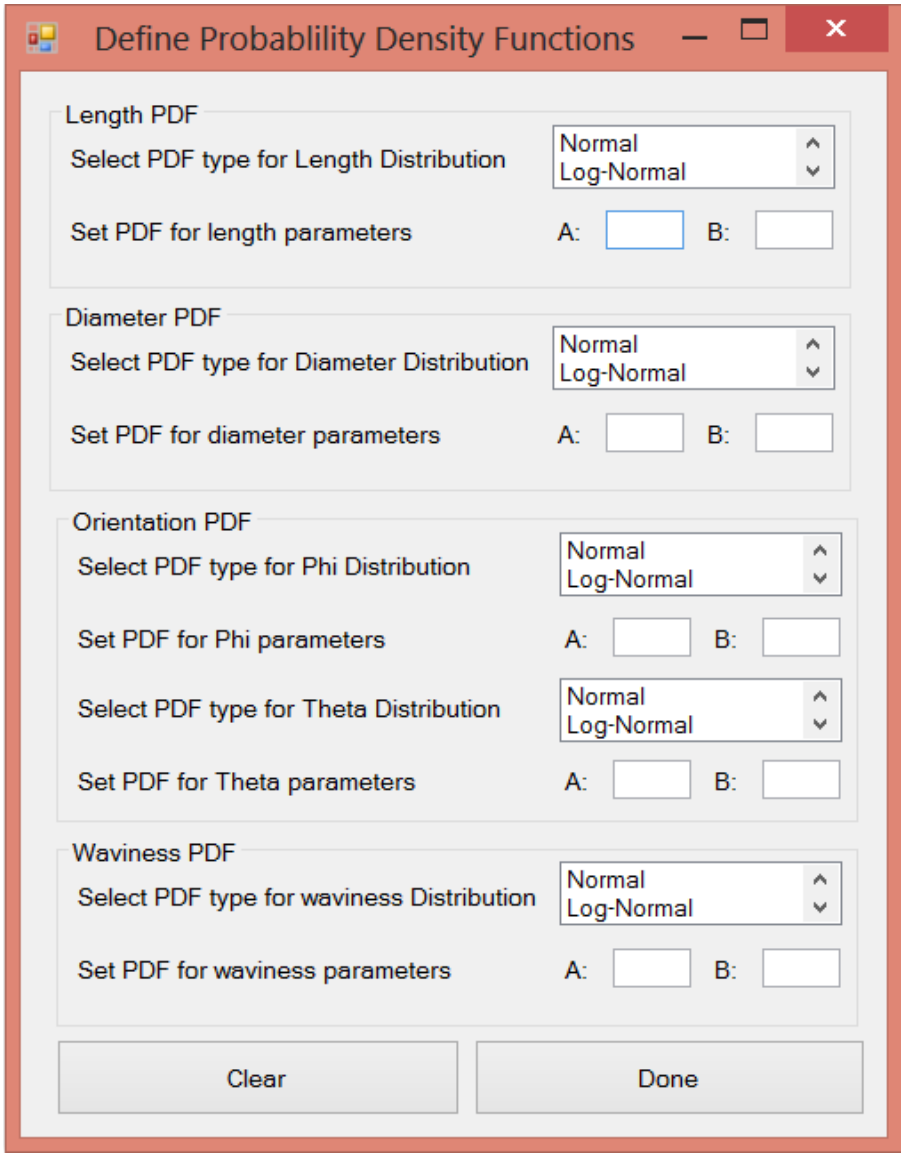

Figure 29: Definition of PDF parameters for the GUI

Again, the values which are used to define the PDF parameters must be obtained from the image analyses discussed in Chapter V. For spherical composites, the GUI screen is slightly altered due to the different inputs and the different readiness level of that particular model.

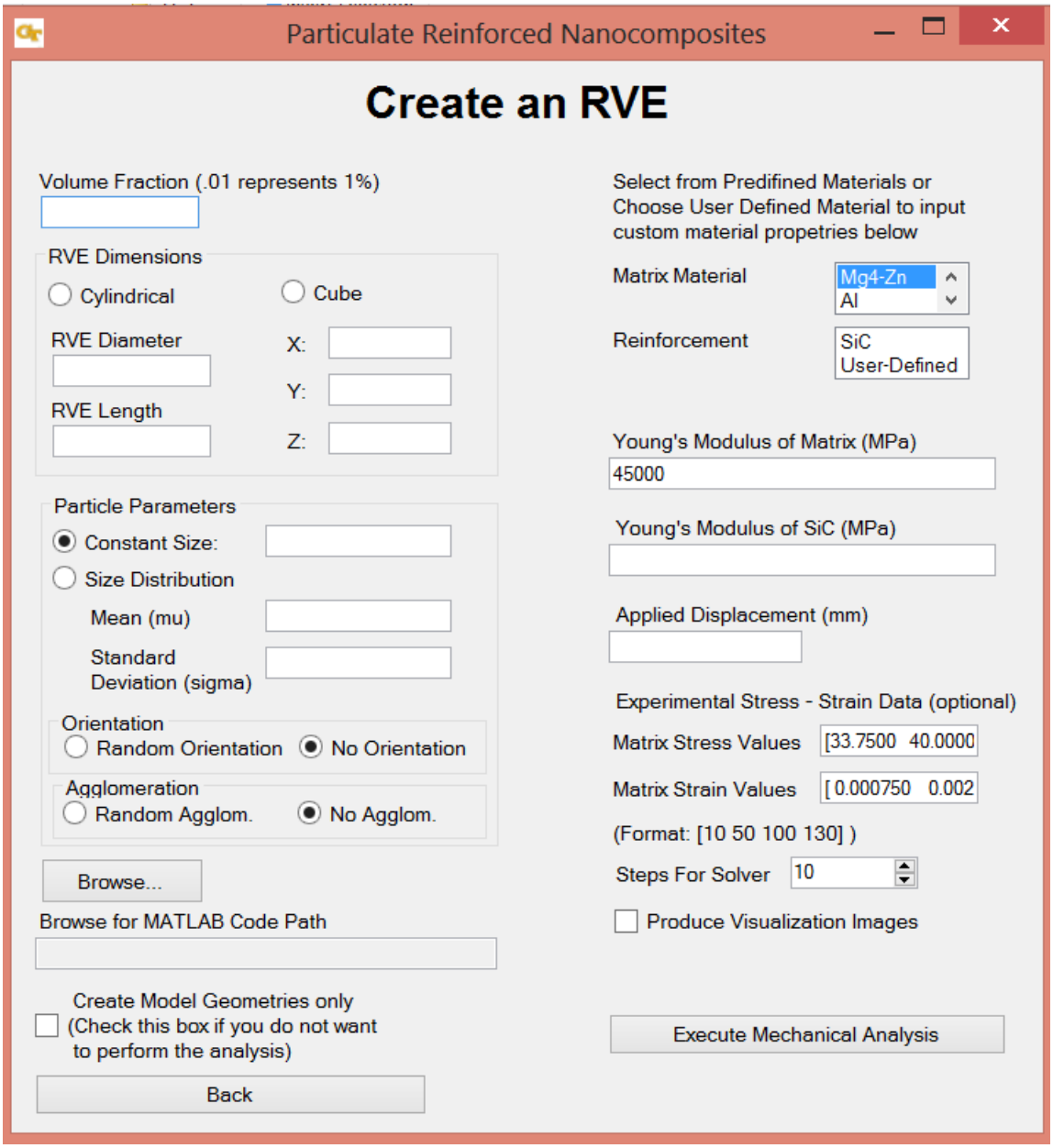

Figure 30: Spherical composites GUI screen

The highlights of the GUI automation include the ease of use for simple and efficient what-if analyses of different nanocomposite materials. The output of the GUI automation tool can be modified using a check box to only produce the 3D models. This has utility for visual assessment of how the dispersion of the filler materials changes with different model parameters. Furthermore, this also allows for the user to obtain the CAD models of the RVE for possible use in external FEA software besides ANSYS. If the analysis is performed, the GUI performs all the steps and then outputs the information both to the GUI and to an excel sheet with the strain, stress, and computed Young's moduli values recorded during the analyses. With this output, this tool presents an easy to use interface to determine the effects of the material parameters such as diameter or waviness on the mechanical performance and properties of the nanocomposite. The GUI also allows for parametric analyses to be performed by sweeping over a range of certain parameters and recording the results. While the GUI tool is in a preliminary state, it is a potentially powerful tool which can aid in the development or analysis of novel nanocomposite materials.

CHAPTER VII

7.1. Applications

The main application of the work presented is as a design tool in the aid of testing nanocomposite material properties. Through validation of the RVE models with experimental composites, it can be determined and found that the RVE models generated could be excellent preliminary estimates as to the performance of their experimental equals. With this validation, it can be justified that the material properties estimated from this design tool accurately represent the performance of the nanocomposite. Two areas are currently being pursued in order to better understand and validate the extent to which the RVE models are currently predicting the experimental performance of the nanocomposites modeled.

7.1.2. Mechanical Analysis Capabilities – Mg4Zn/SiC Case Study

Metal matric nanocomposites, mainly magnesium based, have been extensively studied for lightweight structural applications. As the lightest structural metal, magnesium reinforced with nanoparticles is of interest due to having low density and good specific properties. SiC has been studied as a reinforcement which provides a significant increase in the ultimate tensile strength and the yield strength of the magnesium alloy. A cavitation-based solidification processing technique has been developed which provides a good distribution of SiC particles within the composite [15]. The figure below shows a 3D model of this composite, a meshed model in ANSYS, and a simplified model with the von Mises effective stress distribution in the composite. An important parameter in the properties of this particular composite is the distribution of the particles in the matrix.

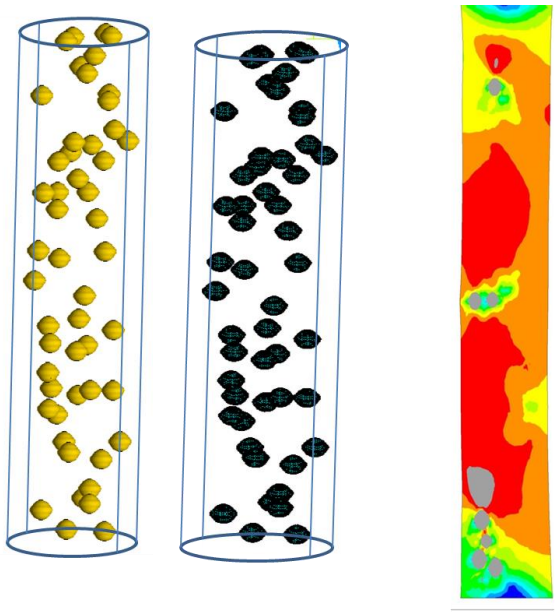

Figure 31: Mg4Zn/SiC RVE model

A cylindrical RVE was chosen for this model because the experimental tensile test occurred on a sample with a cylindrical test section. Once the RVE model has been imported into ANSYS, the boundary conditions are applied as followed: on the two opposite faces along the length of the RVE, the displacement in the x and y directions is set to zero and a displacement is applied equal and opposite in magnitude on both faces corresponding to a given strain that is being tested. The results are taken from the center of the RVE, far from the faces where the boundary conditions were applied. The figure below shows the results comparing the experimentally obtained stress strain curve with the computationally estimated results. In addition, one analytical estimation of the stress-strain curve are shown. The analytical secant modulus is calculated through a method based on Eshelby's concept of mismatch strain between the matrix and the particles [16]. As can be seen the analytical model agrees well with the experimental data. The computational results agree reasonably well with the experimental results. The lower

stresses resulting in the computational model may be due to certain factors which the analytical models account for that are not being accounted for in the FEA method.

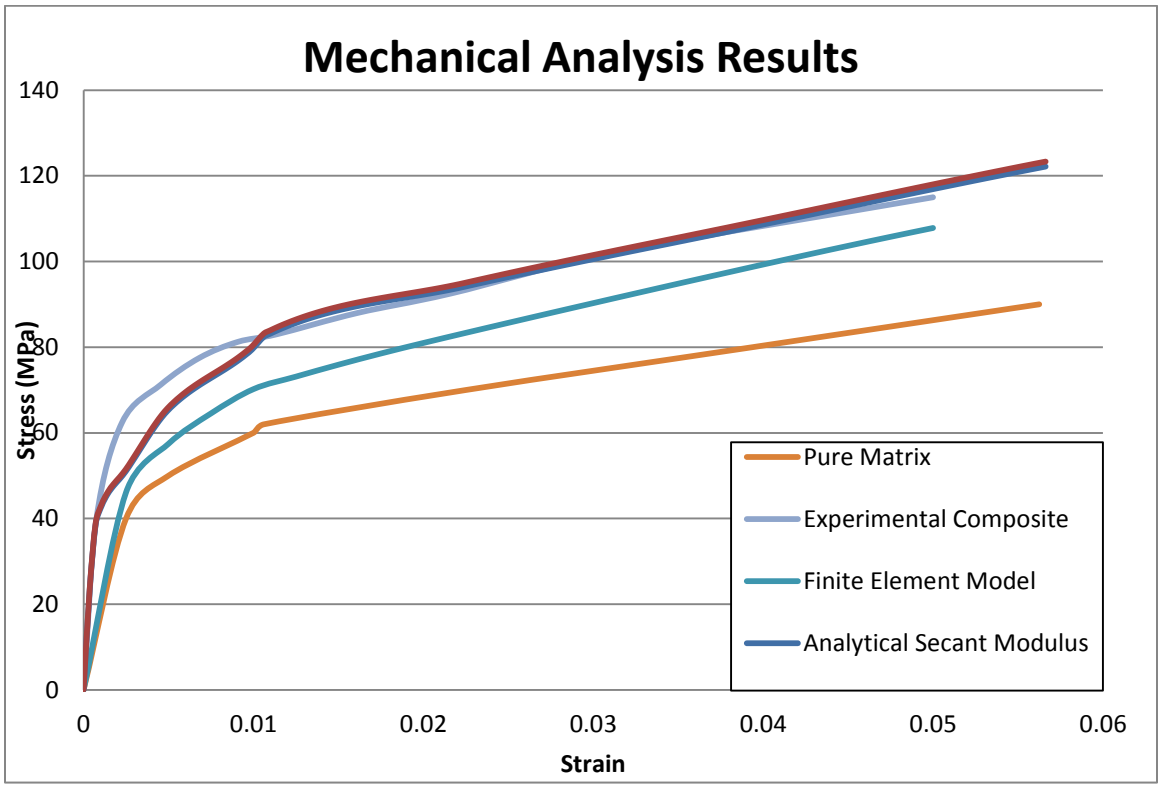

Figure 32: Results from the analysis of the Mg4Zn/SiC RVE

7.1.2. Electrical Analysis Capabilities

Percolation Analysis

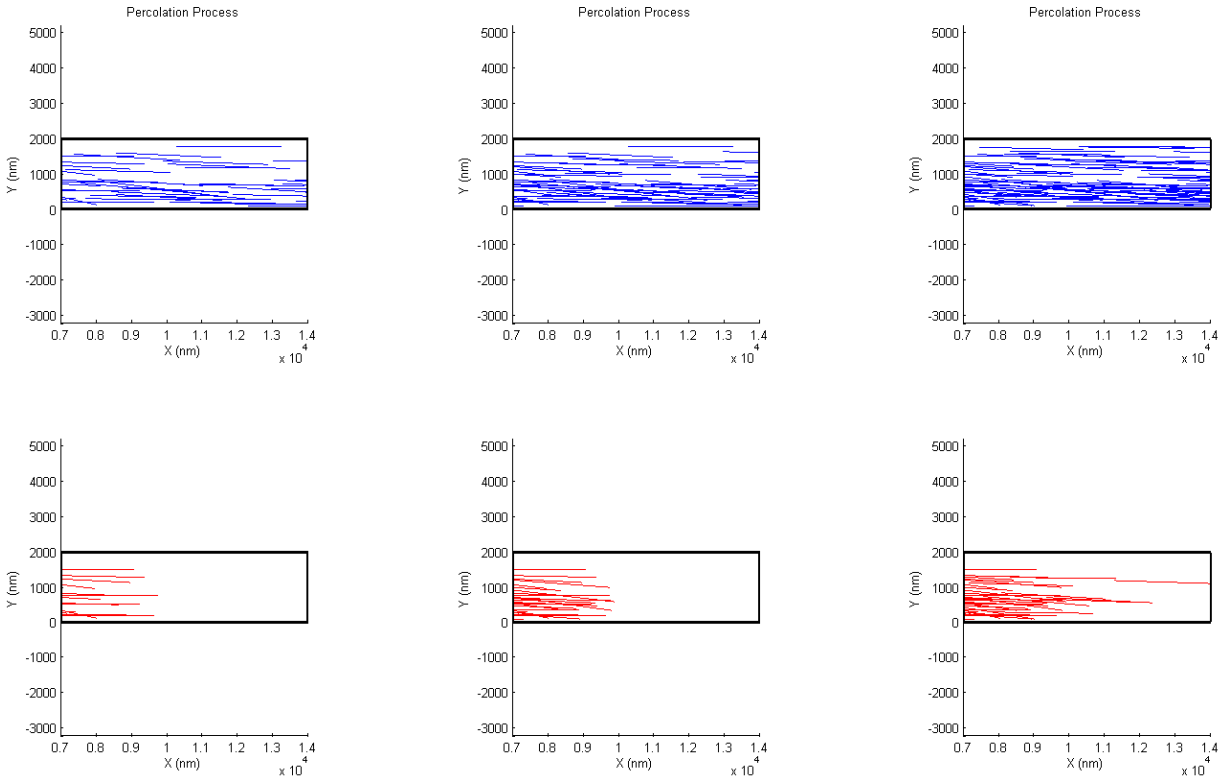

Figure 33: Percolation in RVE at 33%, 66%, and 100% volume fraction

Percolation is understood to be a driving factor for the rise in electrical conductivity of nanocomposites, the percolation of the system should increase with volume fraction along with conductivity. In this study, percolation can be quantified in two ways. In general percolation has happened if there is a network connecting two opposing sides of the RVE. Numerically percolation is quantified by how far the percolating network extends through the RVE, regardless of percolation. The results of the percolation test help to demonstrate the correlation between percolation and conductivity. As the volume fraction increased the length of the percolation network increased as well. At 0.3% volume fraction this percolation network was long enough to extend through the entire RVE. This correlated with the results of electrical conductivity as around 0.3% volume

fraction the conductivity raises considerably; indication the percolation threshold may have been reached. The extent to which the percolation of a network of fibers implies good conductivity can be numerically quantified with the presented model. The same RVE which has been used for a percolation analysis can also be used for an FEA analysis of electrical properties, discussed below, in order to find a relationship between the percolation of a network and its corresponding conductivity.

Finite Element Simulation - PAN/CNT Case Study

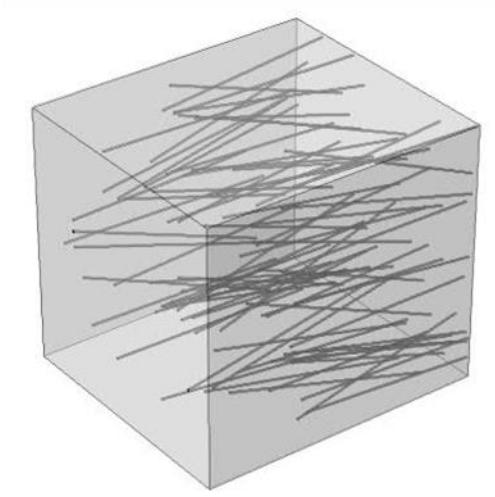

Figure 34: PAN/CNT 3D RVE

Poly Acrylonitrile (PAN)/ CNT composite fibers have been studied recently for use in the conductive textiles industry [17]. PAN/CNT fibers are created specifically to be electrically conductive which could allow for embedded electronics in clothing. Using the appropriate RVE inputs, the figure above shows a model representative of the PAN/CNT composite system which will be used for analysis of the electrical properties of the model. It has been found that the analysis of electrical properties is simpler and more intuitive in COMSOL Multiphysics software than ANSYS. As such the figure shown below is of the meshed geometries in COMSOL prior to the analysis.

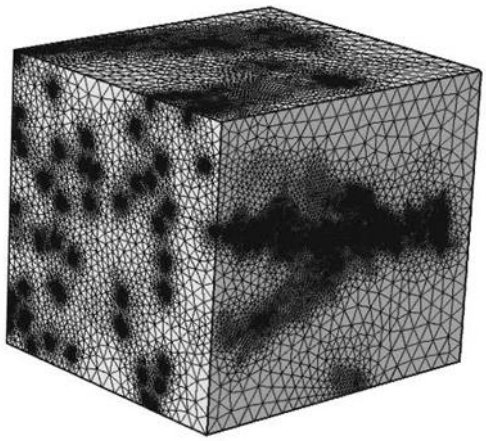

Figure 35: PAN/CNT meshed geometry

The darker areas on the faces are locations where CNTs are close to or touching the face of the matrix. This is an important factor for electrical analysis as if no fiber was to come in contact with the surface, no current flow would exist and the conductivity reported would be unreasonably low. The boundary conditions of the electrical conductivity problem included an applied voltage of 4 Volts on one face and a ground on the opposing face. This generates some current density within the RVE model and this current density is used along with Ohm's law and the dimensions of the RVE to calculate the overall conductivity of the composite.

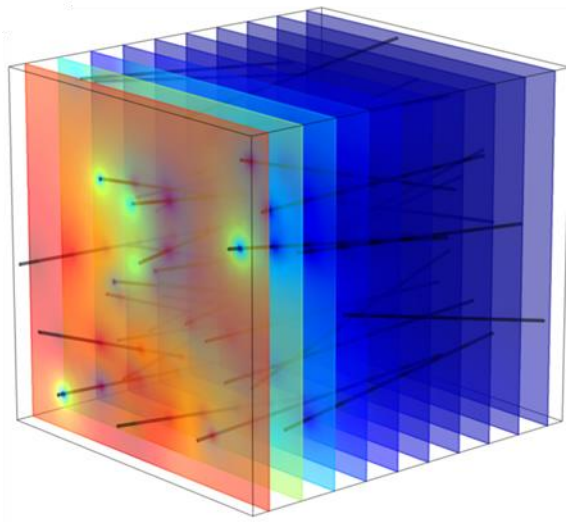

Figure 36: Electric potential in the RVE

Initial results from the electrical analysis indicate that while the general trend of increasing conductivity with volume fraction is evident, the orders of magnitude of the values calculated are substantially lower than in experimental composites. The electrical analysis is more complex than the mechanical analysis and it is believed that there are certain affects which are not being accounted for, such as tunneling current, in the current electrical analysis approach. More work must be done to properly verify the validity of the electrical analysis approach and the values obtained from the analysis.

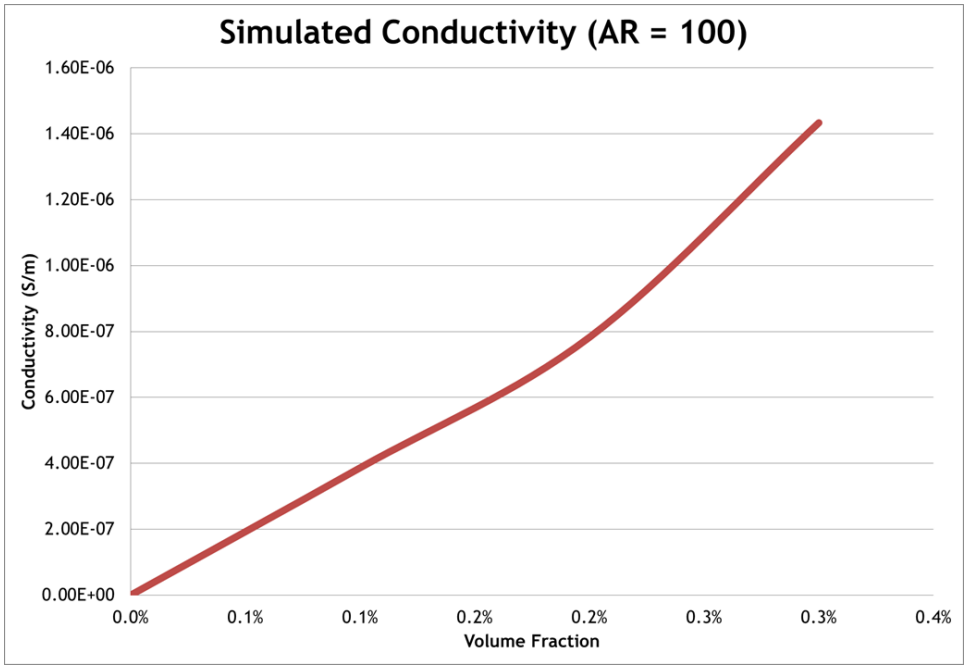

Figure 37: Preliminary results of conductivity

CHAPTER VIII – CONCLUSIONS

3D RVE models for mechanical and electrical characterization of composites with nanofillers are presented. Using image analysis, these RVE models incorporate process parameters and filler morphology in generating 3D network of fillers within RVE. Rod, spherical and disc shaped filler geometries are considered and a filler-to-filler distance algorithm is developed to ensure filler compatibility and interaction within RVE. User interface tools with event driven structures are used to integrate analytical, solid modeling and FEA tools for parametric design and analysis of nanofiller composites with application specific properties. The presented approach uses continuum mechanics to analyze composites with nanofillers. The modular nature of the presented approach however allows incorporation of equivalent continuum modeling of nanofillers using appropriate nanostructure-property relations. Using the presented design-analysis integration, modeling results for stress-strain behavior of MMNC, effective modulus and electrical characterization of PNC fibers are presented and validated. RVE-based integrated models presented are statistical in nature when random process parameters are used to represent the filler morphology and the results are sensitive to RVE size. Monte Carlo simulations may be more suitable if process parameters cannot be quantified from experimental composites. In this work, the statistical nature of results is minimized by using PDF for filler morphology from the SEM image analysis of experimental composites. The RVE models can also be used to analyze continuous CNT reinforced polymer precursors and fibers. The presented multi-cell RVE models enable direct incorporation of filler dispersion, spatial distribution and agglomeration, which is not possible with unit-cell models. However, FEA analysis of multi-cell RVE models with

non-linear material properties can be computationally expensive for higher volume fractions. Overall the developed integrated framework is attractive as a design and quick what-if analysis tool for manufacturing variety of experimental composites with nanofillers for various applications.

CHAPTER IX – REFERENCES

[1] Grujicic, M. and G. Cao (2004). "A computational analysis of the percolation threshold and the electrical conductivity of carbon nanotubes filled polymeric materials." Journal of Materials Science **39**: 4441-4449.

[2] Berhan, L. and A. Sastry (2007). "Modeling percolation in high-aspect-ratio fiber systems. II. The effect of waviness on the percolation onset." Physical Review E **75**(4).

[3] Dalmas, F., R. Dendievel, et al. (2006). "Carbon nanotube-filled polymer composites. Numerical simulation of electrical conductivity in three-dimensional entangled fibrous networks." Acta Materialia **54**(11): 2923-2931.

[4] Berhan, L. and A. Sastry (2007). "Modeling percolation in high-aspect-ratio fiber systems. I. Soft-core versus hard-core models." Physical Review E **75**(4).

[5] http://rosettacode.org/wiki/Closest-pair_problem

[6] Partnership for an Advanced Computing Environment, http://www.pace.gatech.edu/

[7] MATLAB, http://www.mathworks.com/products/parallel-computing/

[8] R. Andrews, M.C. Weisenberger, Curr. Opin. Solid State Mat. Sci., 8 (2004) 31-37.

[9] J.H. Du, J. Bai, H.M. Cheng, Express Polymer Letters, 1 (2007) 253-273.

[10] A.M.K. Esawi, M.M. Farag, Materials & Design, 28 (2007) 2394-2401.

[11] M. Moniruzzaman, K.I. Winey, Macromolecules, 39 (2006) 5194-5205.

[12] M. De Cicco, H. Konishi, G.P. Cao, H.S. Choi, L.S. Turng, J.H. Perepezko, S. Kou, R. Lakes, X.C. Li, Metallurgical and Materials Transactions a-Physical Metallurgy and Materials Science, 40A (2009) 3038-3045.

[13] F. Hussain, M. Hojjati, M. Okamoto, R.E. Gorga, Journal of Composite Materials, 40 (2006) 1511-1575.

[14] G. Vélez-García, P. Wapperom, D. Baird, A. Aning, V. Kunc, Composites Part A: Applied Science and Manufacturing, 43 (2012) 104-113.

[15] G. Cao, J. Kobliska, H. Konishi, X. Li, Metallurgical and Materials Transactions A, 39 (2008) 880-886.

[16]. N. Xu, B.Y. Zong, Computational Materials Science, 43 (2008) 1094-1100.

[17] R. Jain, M.L. Minus, H.G. Chae, S. Kumar, Macromolecular Materials and Engineering, 295 (2010) 742-749.